

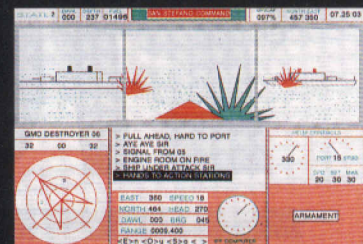
SINCLAIR

QL

WORLD

Volume 2 Issue 7 Price £2.50

CONTEMPLATE YOUR NAVALS! FLEET TACTICAL COMMAND V2.00



SING OUT!
AMIGA QDOS EMULATOR
LEARNS A NEW TUNE.

**MERZ
HYPERHELP**
POINTER ENVIRONMENT AID REVIEWED

ISSN 0951-9335



**Editor**

Helen Armstrong

Publisher

Mark Kasprovicz

Advertising Manager

Jim Peskett

Creative Director

John Stanley

Graphic Artist

Stevie Billington

Magazine Services

Linda Miller, Frances Maxwell,
Pauline Wakeling

**Sinclair QL World,
Published by Arcwind Ltd.
The Blue Barn,
Tew Lane, Wootton,
Woodstock,
Oxon. OX7 1HA
Tel: 0993 811181
Fax: 0993 811481
ISSN 026806X**

If you have any comments or difficulties please write to the editor and we will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.

Back issues are available from the publisher price £2.50 UK, £2.99 Europe. Overseas rates on request.

Subscriptions from: Arcwind
The Blue Barn, Tew Lane,
Wootton, Woodstock, Oxon.
OX7 1HA
UK: £23.40
Europe: £32.90
Rest of World: £40.90

Reprographic Services:
Eclipse, Brook Street,
Watlington, Oxon. OX9 5JH.
Distributed by: Seymour Press
Ltd., Windsor House, 1270
London Road, Norbury,
London, SW16 4DH

© 1993 ARCWIND LTD.
Sinclair QL World is published 12
times a year. All rights reserved.
Reproduction in whole or in part
without written permission is
strictly prohibited. We welcome
contributions. All material must be
supplied with a SAE.

While all reasonable care is taken
in compiling QLWorld, the
publisher and its agents assume
no responsibility in affects arising
therefrom. Opinions expressed
are those of the authors.

VOL 2 ISSUE 7

Contents

8 TROUBLESHOOTER

Bryan Davies looks at LineDesign

10 QL SCENE

Sideways printing ... More membranes ... Pointer programs ... Pemberton programs ... new DJC catalogue ... All Formats

12 OPEN CHANNEL

QLs in Guernsey ... Spectrum Basic .. Debugging commercial programs ... cursor cursed ... more for Z88

14 INSTANT ACCESS

At-a-glance guide to your software and hardware suppliers.

15 QL SCENE

QL-100-Card for control applications ... Show in Italy

16 HARDY HINTS

We only just have time, this month.

18 REVIEW: FLEET TACTICAL COMMAND V2.00

Alan MacTavish is even more hooked than he was with V1.00

20 VERY BASIC SUPERBASIC

Dilwyn Jones makes his choices with the SELECT statement.

23 A FAIR XCHANGE

Simon Goodwin on the new upgraded version of Psion Xchange for the QL

27 PAYMENT POWER

Peter Tomlin uses SuperBasic to track his electricity bills.

31 REVIEW: MERZ HYPERHELP

A new Pointer Environment on-screen Help program tested by Bryan Davies.

33 SUPERBASIC IN ACTION

Simon Goodwin teaches an Amiga Qdos emulator to sing new songs.

36 EASY WITH EASEL PART 4

Henry Orlowski fine-tunes his graph designs.

38 REVIEW: DATA DESIGN

Wolfgang Lenerz looks at the new version of DataDesign

41 CLUB ACCESS**42 MACHINE CODE FOR BEGINNERS - PART 4**

Alan Bridewell contracts conditional branching

46 MICRO ADS**Coming Soon**

We're sorry the Keyword Index isn't here - it's ready but it was left out by accident. Back next month.

TROUBLE SHOOTER

Bryan Davies takes a forward look at LineDesign with Plus4, and scans the scene.

Having finally seen a recent version of Text87 Plus4 working with Linedesign, I realised that the enthusiastic reports of the user who gave the demonstration were clearly not misplaced. Hopefully, we will receive a copy for review before too long. Text87 itself has been changed in several ways, having reached version 3.9, but it looks essentially the same as before, and is still much the same size. The addition of Linedesign to the Print menu is about the only sign of the new-found DTP capability. Most of the changes since previous versions are to the basic WP functions. For example, the editing of the Ruler is now interactive, making it more like Quill, in this respect. This will be welcome to many users; you can see the effect new margins are having on the text. The setting-up of founts for screen display, to match the printed ones, is now done on-screen, and is easier; you can use existing founts lists, too. Again, this is an area where users will have had difficulty with previous versions, and the new approach will be welcome. The Type can be changed several times from within the Block operation.

Linedesign itself is massive, by QL standards, being supplied on ten 3.5-inch DD disks. ED drives or a hard disk drive are desirable devices! There are a variety of detail errors in the version I have seen, such as faulty characters in some founts, but the overall presentation of the program seems generally user-friendly, and the printed results are very good. The instructions are nicely printed, and look to

be well laid-out. The producers of Linedesign have come a long way since they sent samples of their early programs to QL World.

Any DTP-type program relies heavily on the graphical capabilities of the system it is run on, and QL graphics are not fast. Linedesign is quite brisk most of the time, when run on a Gold Card-equipped QL, but there are times when the user will surely wish for the graphics card that Miracle Systems were said to have been developing late last year. Other things - notably the QXL card - got in the way of this project, but it is now reported that Miracle have switched their attention to the card, having finished much of the development work on the QXL. The QXL card may go some way towards speeding-up the 'painting' of the screen, but PC graphics too are decidedly slow.

QSpread

Qspread, the Pointer Environment spreadsheet program has now got to version 1.17, and the man who writes it - Oliver Fink - has written to say he has taken note of the points made in the recent review concerning the user's access to the program. Writers of application programs are sometimes placed in a difficult situation by the operating system and/or supervisory interface through which their programs communicate with the computer, and this is certainly the case with QSpread. My sympathy is with the programmer. Whether or not there is something intrinsic to the "graphical user interface"

concept which slows down the user's connection with the application program is a very debatable point. It is certainly true that the Windows GUI on the PC has this effect. On the other hand, it is also true that it takes application programmers some time to find out ways of getting GUIs to perform, and later version of programs can be appreciably faster than the initial ones.

Ramdisks

Reader Michael Hussmann, from Hamburg, has taken me to task and requested that I make amendments to comments made about ramdisks in a previous article, but further checks on my QLs do not suggest the points made earlier were wrong. The reference to ramdisks came in connection with using Quill, and trying to prevent it commandeering all the available free memory. One way to restrain Quill would appear to be to format a ramdisk of sufficient size to reserve all the memory Quill really does not need, before Quill is loaded, the idea being that you could then get rid of the ramdisk, freeing the memory for use by other programs. The problem as I saw it was that the static ramdisk routines, as were supplied with some interface boards, or as separate software, years ago would do the job simply, but the dynamic ramdisk routines which are now normal would not.

On a Gold Card-equipped QL, the DIR RAM1_ command will show a capacity of roughly the amount of memory currently not in use (use the TK2_EXT command first). This ram drive is dynamic in the sense that the

memory it is nominally occupying will be given to programs, as they require it, so starting Quill results in all the spare memory being taken by it (all bar 4608 bytes, according to the PRINT FREE_MEM command). Attempting to get around this behaviour by specifically formatting a fixed-size ramdisk, got me nowhere either, since reducing the size of this 'drive' to zero by giving the command FORMAT RAM1_0 after Quill had started left me unable to run even a mini-program, and the FREE_MEM figure was still 4608 bytes. The same procedure worked quite satisfactorily with non-Psion programs, with the memory reserved by the ramdisk becoming available to any programs started subsequently.

At first sight, that was the end of the story but, in a subsequent discussion with Freddy Vachha of Digital Precision, he suggested making use of the NEW command, before and after the FORMAT command, just in case that freed-up the memory that a simple FORMAT RAM1_0 had not. It was an obvious point - once Freddy had reminded me of it - and it did the trick. Unfortunately, it worked once, but nothing I did subsequently reproduced the same result. The Trump Card would be likely to give similar results. Rather than spend a long time trying to find a fool-proof way of reserving memory this way, it should be more effective to use the commands ALLOCATION and DEALLOCATE (Turbo Toolkit) and ALCHP, RECHP and CLCHP (Toolkit 2), or use Simon Goodwin's Taskforce.

Minerva

My comments about the Minerva rom clearly upset TF Services, and some clarification is needed. My feeling about modifications to QDOS is not that they shouldn't be attempted, but that existing software should be taken into consideration from the start. People who delved into Qdos and SuperBasic years ago found errors, and wrote programs to get around the errors. The QL has developed a lot over the years, without Qdos being touched, as witness the excellent collection of hardware and software we have now that did not exist a few years ago.

A complete new operating system, which is what Tony Tebby is said to have been working on for some time, is a different matter. No-one is better placed than Tony to know what needs doing to improve upon Qdos. Presumably, we have seen, and will see, elements of the new OS in the Atari and QXL hardware QL emulator cards. It is to be hoped that the new system will be able to run all existing, major software and does not produce a crop of complaints about incompatibility to vex hardware designers and software publishers alike.

(Next week, jousting at Windsor-Ed.)

And now - incidentally - another satisfied Minerva customer. Geoff Wicks responded to comments made in this column about the use of the Minerva rom. He reports finding no significant compatibility problems when using his regular software (Psion PC-Four, Perfection SE, Professional Publisher, Conqueror SE, Eye-Q, FileD) on a Minerva-equipped QL, but he has had some trouble using Front Page Extra and Flight Simulator. He has managed to "bodge" FPE, using a routine supplied with Minerva, but the same fix produces "the most impressive crash on a QL that I have yet seen." Digital Precision's Lightning is in use on his QL most of the time.

He did point out that he has a relatively-recent version of Minerva - 1.93 - and that he had had crashes when using a Trump Card in the same system,

but is happy with the behaviour when a Gold Card is fitted. A couple of his own public-domain programs which gave trouble on a standard JM rom QL work fine with Minerva. He cites the features of Minerva that are of most advantages to him as the composing of accented characters, better editing with the EDIT and INPUT commands, and a warm reset which does not corrupt the Gold Card clock.

Readers' Letters

I M Gaye sent in two 3.5-inch disks, one DD and one HD. He could not get either of them to format to the full capacity, and asked for a second opinion on them. He has both DD and HD drives. The HD disk was no problem - it formatted to the full capacity first-try, on both my QL and PC. This suggests there is a problem with the HD drive Gaye used to format it. The DD disk was another matter. After more than ten attempts, on both QL and PC, it gave no more than the capacity that Gaye had obtained from it. Various attempts to "doctor" it did not increase the capacity. This was not surprising, as a visual examination of the disk surface revealed a wear ring near to the disk centre, which was the area reported by software as being bad. Something had damaged the disk; either the drive read/write head had scraped it or some dirt had got in and done the job. As a final check, an HD format was attempted in an HD drive, and the disk was immediately branded as being unusable for this. This may not seem surprising, but it is most unusual, in my experience.

One thing to bear in mind when using drives to format disks of different densities is that you can get odd results; for example, one of the tests with these two disks produced 720/720 sectors from a disk that should give 2880/2880. If you are using a Gold Card, make use of the command FLP_DENSITY X, inserting either D, H or E (for DD, HD or ED density) for X, as appropriate. This ensures that the interface software knows what the disk to be formatted is.

JOCHEN MERZ SOFTWARE

Software - Development - QL-Soft- & Hardware Distributor
Im stillen Winkel 12 - D-4100 Duisburg 11 Germany
Telephone & Fax: 0203 501274 • Mailbox 0203-591706

New Games: The Oracle (by Jochen Merz) The Oracle is an ancient tactic-puzzle, where you do not have to be fast but clever! You have to fill different tiles into a field, but there are various rules how to do it. If you can place all the tiles you a bonus - if you obey special rules you get much more points. Every game is different, you'll never be bored. Features: high-score table, hints for the next move etc. **DM 49,90**

MineField (by Bernhard Scheffold) Another game for the Pointer Environment. You need skill and concentration to clear a minefield. Many options, configurable size, number of mines etc. Toolkit II required. **DM 39,90**

QD Version 5 - The first (and only) Editor using the PE. Dynamic memory allocation, no limit on numbers of lines, comfortable block-handling and many, many new features, e.g. improved print menu, better search/replace, GOTO Procedure and function, even Machine code label.

New features: VS Thing Interface which makes QD5 extendable and inbuilt **HyperText HELP System!** This HELP System is very easy to use, simply move the cursor over a word and press HELP (F1) and you get extensive help on the subject. With complete HELP (German and English) for SuperBASIC, inclusive examples! It is very easy to make Assembler-Help, e.g. get external definition of routines, libraries etc. with one keystroke. HELP can be called recursively, which means, you can get help on previous help on another help subject. You can even edit the help texts, add remarks, examples etc. **DM 125,-**

Upgrade from QD V4 DM 30,-

All our products which contain SuperBASIC extensions will be updated so that they will have files which you add to your HELP System so that help is provided for additional Procedures/Functions!

HyperHELP BASIC

This product gives you instant help in SuperBASIC! The price is much lower for those who program in SuperBASIC only and do not require a full QD (although it is very useful). HyperHELP can be executed, put onto HOTKEY or woken from a Button. It displays the full set of SuperBASIC procedures, functions and keywords currently existing in your machine, plus additional help on operators, identifiers etc. Simply click on the word you get the full description, use of all parameters plus examples. **FORGET ABOUT YOUR MANUALS** for parameter description! You get complete help on SuperBASIC, in German and English. The help files can be updated with any editor, Quill, Text87 so that you can update and add remarks whenever you want. **DM 49,-**

QMenu-the Menu Extension NOW IN VERSION 4!

QMenu is a very easy to use interface with pre-defined menus (e.g. multi-column file-select, simple-choice boxes, select from lists, error handling). These menus may be used from SuperBASIC, machine code and other languages. New features: the directory name is (optionally) not fully listed, i.e. only the names INSIDE the subdirectory are given, allowing for much more files to be listed in the window. New **DO AND REPORT** menu. Brandnew feature: pre-defined directories and Extensions may be changed and configured at runtime! More examples, improvements here and there, which makes the menu Extension getting mature. **DM 39,90**

Update with new manual DM 16,-

QSpread - a new Spreadsheet program. It is completely mouse- and/or keyboard-controllable and uses, of course, the Menu Extension. Windows may be enlarged up to the maximum screen area, it may be split in up to three different horizontal and vertical sections, giving 9 independent controllable areas. Every cell may be formatted independently, with many options (justification, decimal point etc.) and with preview. Sum- and other often-used macro-functions. Maximum: 32768 cells, where, for example, 16000 cells need about 400kBytes. Block handling and block entry is very easy, especially if you have a numerical pad: you select the block and enter the values one after the other. They are automatically place in the right order. No cursor-key-action necessary! New cellname-enquiry, echo-function, different rounding methods. Many additional functions, which belong to today's standard-equipment: Help, Button-function, use of the Scrap, all standard file operations, calculation order row or columns etc. [V1.12] **QSpread with comprehensive manual DM 169,-**

QDOS Reference Manual DM 89,90

SER Mouse driver DM 40,-

Fifi DM 49,90

QShang DM 45,90

LineDesign DM 249,-

Diamonds DM 35,90

Arcanoid DM 35,90

SuperGamesPack DM 90,-

EasyPTR Part 1 DM 89,-

OPTR DM 92,-

QSUP DM 79,90

DISA DM 90,-

Thing & EPROM Manager DM 61,50

text87plus4 DM 239

BrainSmasher DM 45,90

Firebirds DM 35,90

The lonely Joker DM 52,90

Part 2 DM 49,- Part 3 DM 49,-

QVME - Emulator card for ATARI Mega STE DM 695,-

Harddisk-Kit for ATARI Mega STE DM 99,-

Medium for 44MB SyQuest changeable harddisk DM 143,-

All sorts of Quantum SCSI-harddisks at cheap prices available, spare ones or readily cased (with cables, adaptor etc) for ATARIs.

Please add **DM 13,-** for postage and package (Europe) or **DM 13,-** for one item and **DM 7,-** for every further item (Overseas). All prices incl. 15% V.A.T. (can be deducted for orders from non-EEC-countries). E&OE.



Keyboard membranes ahoy!

If your QL's keyboard isn't behaving like its old (or young) self, we have some good news. Bill Richardson of EEC has negotiated a whole new batch of **membranes** from the manufacturers. (The membrane is the bit that goes under the keys, and wears out.) The catch is that he had had to order a good-sized batch, so if you think you're going to need a new keyboard membrane in the near future, now is the time to let Bill know. If you aren't sure, call Bill, or Tony Firshman at TFServices, and they can, no doubt, give you some advice.

The new membranes cost #10 one-off, but there are discounts for quantities over 20.

The deal included new membranes for Spectrum models Spectrum+, Spectrum+2, -3 and 128K, so if you know anyone who has an ailing Spectrum, tell them the news.

EEC's **Universal Disk Drives** have spawned a new, cheaper version. The **Economy Universal Disk Drives** do away with all the external DIP switches, connectors etc. that are not needed, and are suitable for the QL, Spectrum and PCs only.

They come in 4MB, 2MB and 1MB capacity. You do not have to buy a pair of identical-capacity drives: you can mix-and-match to suit your needs, and save more money. The 4MB drives cost #60 each, the lower-capacity drives are less.

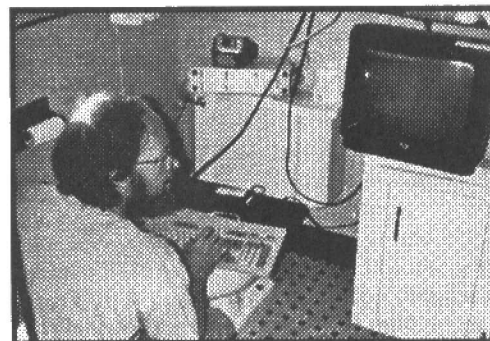
The fully-configurable original Universal Drives are also still available. Ask Bill for advice.

WNR Richardson & Co. (EEC), 18-21 Misbourne House, Chiltern Hill, Chalfont St. Peter, SL9 9UE. Tel. 0753 888866; TFServices, 12 Bouverie Place, London W2 1RB. Tel. 071 724 9053.

What, no Chairs?

Bill Richardson reports from the QL meeting held by IQLR at Newport, Rhode Island, USA on 5th June! Business was good, says Bill. His photos show

Tony Firshman of TF Services "closetted" with his QL set-up in the toilet of the Motel where the UK party stayed, getting ready for the latest Minerva demonstrations at the show. We've heard of putting the QL on a pedestal, but this just goes to show that you can take your favourite computer anywhere! Let's hope Tony was on a roll ... "Feature of the meeting" was Stuart Honeyball and the new QXL Interface. Miracle is now IQLR's agent in the UK - a true transatlantic symbiosis.



Pointer Programs for Mouse Maniacs

Following last month's review of the Albin Hessler mouse interface by Bryan Davies, QL Scene draws readers' attention to some of the range of programs written to run under the **QJump Pointer Environment**, a mouse-driven front end and management program for the QL. The Pointer Environment itself is included with most major programs designed to run under it.

Dilwyn Jones Software (tel. 0248 354023) has an agreement with **Care Electronics** to market most of their QL products. Software for the Pointer Environment includes **Qpac1**, **Qpac2** and **Qtty**. **Qpac1** is a set of Pointer Environment utilities, which include as part of their number a pop-up calculator, calendar, alarm, typewriter utility, digital clock and Sysmon (system monitor) utility. The Pointer Environment is included. On disk only, for expanded memory. **Qpac2** is a full file maintenance program, allowing copy, delete, view, format, backup, move, execute, print, channels information, hotkey listing, jobs menu, Things menu, Sysdef utility, program-picking utility (to list programs currently running, for you to select - saves repeated Ctrl-C'ing). **Qpac2** includes a tutorial program. Disk only, for expanded QL memory.

Qtty is a spelling and typing checker which can be used with most QL programs, including Quill. Also supplied as part of text87 Plus4, it comes with a 40,000 English dictionary, plus French and German dictionaries, and a dictionary editor. A SuperBasic interface allows you to write Basic programs that use Qtty. On disk for expanded memory only.

These three programs are written by Tony Tebby, and Dilwyn Jones of DJC is happy to give advice on request.

Jochen Merz Software (010 49 203501274, Germany) has a long list of programs using the Pointer Environment. Merz products usually come with the Extended Environment, including the latest Pointer Interface, window manager, Hotkey System II and standard CONFIG. QD and QSpread have a menu-driven CONFIG. The minimum requirement is usually a Qdos machine (QL or Atari ST/QL emulator), one disk drive and ram expansion. Most programs are Minerva-compatible, but the Thor remains a mystery.

Merz's roster has on it **QD**, the Pointer Environment text editor; **QSpread**, the PE spreadsheet with up to 32768 cells; **QSup** system utilities - system defaults, translation table editor, notepad, printer panel, etc.; **Fifi**, the menu-driven file finder; **QDesign**, a graphics program with virtual graphics on QL emulator and Minerva, up to 2880x2880 pixels, choice of screen/page standards, etc.; **Vectedit** vector font editor for QDesign's scalable fonts; **DISA**, the intelligent disassembler; **DataDesign** advanced database; **QPTR**, the programming manual for the Pointer Interface, with SuperBasic extensions to allow access to the facilities; **EasyPTR** menu-driven menu designer, and **QMenu**, which allows the use of the Menu Extension in your own Basic and machine code programs.

Merz even do three games for the Pointer Environment: **Brain Smasher**, **QShang** and **The Lonely Joker**.

If you are buying software for a mouse-driven system, check your supplier's opinion on whether the software you want will work with your mouse interface. With newer mice, the answer should be positive. Some people hate mice, but others become addicted.

Printing Sideways

DJC in action again - their new one is called **Sidewriter**, a sideways-printer utility which prints spreadsheets and text files sideways across the length of a sheet of paper. Sidewriter was partly inspired by a letter in QL World's readers' column Open Channel, March and May 1993. "This program was planned before those letters in the magazine," writes Dilwyn Jones, "but they speeded up its completion."

Sidewriter takes exported or printed Abacus spreadsheet files, or any plain text file. The program can be either pointer driven with a mouse, or used from the keyboard without the pointer environment installed (for unexpanded QLs, for instance). It can be used with QSpread with a plain text filter or a file printout.

Lines printed sideways can be configured at 256 characters as supplied, or even longer lines. It offers a choice of 6,8,9, or 12 lines per inch across the paper, handles paper up to 16 inches wide, and up to 2048 characters sideways down the paper. There are 17 fonts in two sizes, and all may be printed in normal or condensed width, giving a choice of four character-widths in all.

Gridlines can be printed to guide the eye along columns of figures in large spreadsheets.

Expanded memory is required if the pointer environment is in use. The PE is supplied with the package; if used without it, Sidewriter will run on an unexpanded QL.

Sidewriter costs £15 disk or microcassette, including UK postage. Add £1 for overseas postage. Order from Dilwyn Jones Computing, 41 Bro Emrys, Tal-y-bont, Bangor, Gwynedd LL57 3YT. Tel. 0248 354023.

CASH FLOW

	January	February	March	April	May	June	July
SALES	4000.00	4080.00	4161.60	4244.83	4329.73	4416.32	4504.65
COST OF SALES	2750.00	2790.00	2830.80	2872.42	2914.86	2958.16	3002.32
GROSS PROFIT	1250.00	1290.00	1330.80	1372.42	1414.86	1458.16	1502.32
EXPENSES							
wages	700.00	700.00	700.00	700.00	700.00	700.00	700.00
advertising	100.00	100.00	100.00	100.00	100.00	100.00	100.00
rent	200.00	200.00	200.00	200.00	200.00	200.00	200.00
electricity	50.00	50.00	50.00	50.00	50.00	50.00	50.00
depreciation	90.00	90.00	90.00	90.00	90.00	90.00	90.00
TOTAL EXPENSES	1140.00	1140.00	1140.00	1140.00	1140.00	1140.00	1140.00
NET PROFIT	110.00	150.00	190.80	232.42	274.86	318.16	362.32

New DJC Catalogue

DJC has produced the latest issue of its 20-page **catalogue**, detailing a range of about 100 QL products, from microdrive cartridges to software. It includes descriptions of all items, including many brand new releases, as well as sections listing addresses and telephone numbers for QL suppliers worldwide, and answering some commonly-asked QL questions. New products listed include the serial mouse system for the QL, Sidewriter (a utility for landscape printing of wide spreadsheets or other documents), and a range of software formerly published by CGH Services before their closure in March 1993.

What Dilwyn doesn't say in his press release is that the catalogue contains dozens of good, low-cost programs, and a number of important less-low-cost ones as well. Don't overlook DJC if you are expanding your software base.

Dilwyn Jones Computing, 41 Bro Emrys, Tal-y-bont, Bangor, Gwynedd, LL57 3YT. Tel. 0248 354023.

EX-CGH ADVENTURES

Alan Pemberton has made his popular adventure games into SQLUG-ware.

Formerly published by CGH Services, who closed down earlier this year, the five adventures are on a two-disk compilation called "adventure93. Adventure Playtime, the Tower of Valagon, Starplod and Ye Classical-Type Adventure are on Disk 1. On Disk 2 the Voyage of the Beano stands alone.

Access to SQLUG is on two levels. Adventure93 can be obtained from Qubbesoft and copied as public domain so long as all the accompanying documentation is included. In addition, users may register their SQLUGware package for a fee (in this case for £7), for which they receive printed manuals, a helpline service, and free upgrades when they appear.

SQLUG (the Scottish QL Users Group) would like more people to know about their activities and contact them. For information or to register your SQLUGware, write to Alan Pemberton, SQLUG Secretary, 65 Lingerwood Road, Newtongrange, Dalkeith, Midlothian EH22 4QQ.

ALL FORMATS DIARY

Coming dates for the All Formats Computer Fair are:

J25 Jul West Midlands National Motorcycle Museum, junction 6 of the M42; August: no dates listed; Sep West Midlands National Motorcycle Museum, junction 6 of the M42; 12 Sep Brighton Corn Exchange, Church St, Brighton 18 Sep Leicester De Montfort Hall, Granville St, Leicester.

Check with suppliers whether they will be at a particular Fair. If you have far to travel phone All Formats 0608 663820 to check arrangements haven't changed.

Day tickets are £4; attendees can get up to 50 £1-off vouchers by sending an SAE to the organisers at Maple Leaf, Stretton-on-Fosse, Moreton-in March, Gloucestershire GL56 9QX. (Only one voucher per ticket.) Photocopies of these vouchers are also accepted. Admission to the Fairs is a flat £2 between 2pm and 4pm (£1-off vouchers do not apply at these times).

QL Scene

OpenChannel

Guernsey calling Bug fixes

I had a response from a man in Guernsey to the letter you published in the March issue. Over the 'phone he said he had thought he was the only QL user in Guernsey until he read my letter in the magazine.

After talking with him, I started thinking more about the future of my two QLs and my two **Psion Organisers**, and totted up all that I have spent on them. My thinking was, before the conversation, that the amount of money needed to improve my QL and Organisers would be spent more sensibly on an obsolescent and therefore cheaper 386 and a Psion series 3. However, this would mean all I had spent (and done) previously would be wasted.

Now, though, if I decide to spend money on upgrading, I will improve my QL D16 with a Gold Card and two ED disk drives. My QL D11 will be improved, one day, to be more like a D16 (I started on it about three years ago!) The Organisers are being improved with 2*256K Flash ram packs. And I will continue to improve in my use of things I have learnt a little about over the last seven years instead of having to start again with a new set of conundrums like I had to when changing, after eight years, from one of the early American 8K PETs, later expanded to 32K!

I have bought a book which I hope will enable me to learn how to communicate better between the QL and the Organiser. Perhaps I will learn how to use the Tandata communications unit more effectively with the QL I retire, at least partially, next year, so I should have more time.

Lawrence Carpenter
Guernsey
Channel Is.

May I also correct an impression? The incompatibility problem mentioned by Tony Firshman in last month's **Open Channel** was referred to me by another user, and has since been confirmed. It has apparently been cleared up, as I have not had any further reports. What I said to Tony, in the discussion he mentioned, was that I did not remember the details at that time (in the middle of a Quanta meeting), but there was no problem about checking it.

We have seen something of a new operating system in the Atari QL emulator, and now in the QXL card, and both of them are, to me, welcome developments. However, if any new OS causes significant problems with existing software, it will receive complaints from users, and some complaints are likely to be passed to me.

Bryan F Davies
Kent

I read with interest the review in Vol. II No. 5 of QL World of QMaths. As I am the author of one of the programs on the QMaths disk, **Q-Fract**, I gave particular attention to this part of the review, and I would like to thank the author for bringing to my attention a couple of small problems in 1.06, both of which I have corrected in 1.07.

The first problem required me to borrow a QL with an old "slow" memory expansion, as I have a Gold Card system. I could not help but agree with the reviewer that the old standard directory routine was a bit slow, so I have rewritten it. The new version is about five times faster on most systems, and so should be crisp enough to keep most users happy.

The second problem, inexplicable error messages, was more

awkward, as it only occurred with **Minerva** systems, and I had difficulty gaining access to such a system to do testing. The result was that I found that the fault in the **Minerva**, which sometimes does strange things when comparing strings. In spite of this, I altered my program so that the offending errors should no longer occur.

I would like to take this opportunity to make a **plea**, both to **Minerva** users and to author Lawrence Reeves. When a program does not work properly on systems fitted with a **Minerva** rom, I should like to know that a process such as the following takes place:

The author of the malfunctioning program checks it for bugs, and corrects any found. The **Minerva** programmer also checks his code for bugs which (since he is human) it may well contain, and if the fault lies with **Minerva**, it can be corrected in the **Minerva** source code. Communication may sometimes be called for between the programmer and other software authors, and both parties should be willing if need be to establish contact.

Unfortunately, this is not what often happens. Almost every time a program misbehaves on a **Minerva** system, the assumption is that **Minerva** cannot be at fault. This is frankly unlikely as, brilliant though he is, Lawrence too must make mistakes. My contact with QL users and software authors suggests that many (possibly most) of the problems are not even brought to Lawrence's attention, since users assume that **Minerva** cannot be to blame. If **Minerva** is to evolve harmoniously, this must change, and where a program misbehaves only on **Minerva** systems both the author of the program and the author of **Minerva** may have work to do.

When I examined the prob-

lems identified by your reviewer, I was able to correct my code within a few hours. Lawrence Reeves is certainly a better programmer than I am, but for goodness' sake give the man a chance! If a package won't work properly on your **Minerva** system, report the problem in detail to your **Minerva** supplier as well as to the supplier of the other program. That way is the greatest likelihood that it will be put right wherever possible.

Mark Knight
Notting Hill
London

*It's funny how these things suddenly sit up again after years of quiet, isn't it? Is it the weather? There's no doubt that when **Minerva** first appeared, some adherents took an aggressive attitude towards anyone who denied that **Minerva** was the One True Rom. This more than anything else created the "Minerva question", which need otherwise never have existed. Oddly, I never received a reasoned defence of the position from any user, although we discussed it a good deal internally, having both the supportive and the sceptical on our team. Eventually after some correspondence the QView team themselves offered us an article on **Minerva's** rationale and history, and I eagerly accepted, but it never came to fruition.*

*In the intervening years, more usefully, work has been done both on **Minerva** and on other software to improve inter-relationships. However, I think that Mark is basically right, and that it is still widely assumed that **Minerva** is either intrinsically right, or intrinsically intractable. Neither of which is true. If you have what looks like an incompatibility problem, talk to both publishers. And let the best man refrain from treating differences as Sins.*

Speccky Basic

The following information might be of interest to some of your correspondents. **Spectrum Basic** is a superset of ZX80/81 Basic. In other words, the two are alike, but the Spectrum version contains extra commands for colour, sound and graphics, plus a hook code for paging the Interface 1 shadow rom when any of its routines are invoked. Although QL SuperBasic contains functions and procedures that would be familiar to Spectrum users, there are a number of differences. It was written from scratch, ie the actual code is not derived from another product; the single-key command entry system was not implemented, and the file-handling commands use a totally different syntax. For example, in Spectrum Basic:

```
LOAD "M";1;"MYPROG"
```

In SuperBasic:

```
LOAD MDV1_MYPROG
```

Tony Tebby, being the sensible fellow that he is, decided that he wouldn't subject QL users to the vile, sadistic and brutal form of torture known by PC, ST and Amiga owners as the Command Line Interface. To put it another way, you have to communicate with QDOS via machine code or a high-level programming language - which is one of the reasons why SuperBasic was constructed as a distinct and separate entity.

You can turn your monitor into a tropical fish tank by purchasing **Screen Dazzler**, a wonderful new product from Dilwyn Jones Computing. However, strongly advised that you do not pour water or fish food into your display, since neither material is QDOS- (or electricity-) compatible.

K J Brickwood
Stockwell
London

Faint cursor

For nearly eight years, my trusty old QL has given me first class service. Problems have been few and have generally been sorted out by reference to my collection of QL Worlds, books or by harassing a computer whiz-kid friend of mine. Recently, however, I have come across something which I haven't been able to solve and would be grateful for tips.

My Microvitec Cub monitor has been a bit wobbly (an internal fuse blew, which I replaced. Then several months later the sides of the image on the screen started to bow inwards). At the moment I can't afford to have it looked at, so I am using a Philips amber screen monochrome monitor. The program I use, almost exclusively is The Editor Special Edition, and the problem is that, although the Paper and Ink colours can be readily set to give good contrast on a black-and-white screen, there does not seem to be any

way to change the colour (or character) of the **cursor**, which is very faint.

I assume that the cursor in the program is the standard QL cursor. Looking at my reference books on Qdos by Colin Opie, Andrew Pennell, Jan Jones, etc, it is apparent that, while there are routines to toggle the cursor on and off, move it up and down and position it on the screen, there does not seem to be any way of altering the colour or shape of it. Please put me out of my misery: can it be done or can't it?

Gwynne Owen-Smith
Ashford
Kent

Freddie Vaccha of Digital Precision, publishers of The Editor, answers: "Cursor parameters are not changeable on a Sinclair QL, unfortunately. For example, cursor colour is always the XOR of the underlying strip colour (so red gives black, white gives green, etc.). There is a way to edit the cursor, but alas

only on Minerva QLs." (Which involve something of a change of QL culture.) "Hermes, from the same stable, is eminently recommendable, but doesn't help with cursors."

Thanks to Dilwyn Jones of DJC, we have some more **addresses** which will be useful to Sinclair Z88 users. Dilwyn writes:

"To help Jean-Yves Rouffiac with his dead Z88, here are some adverts by companies who still support the Z88, including the current address of the Z88 User Group. He should be able to find someone who can repair the machine for him.

"He may also like to know that there are at least two utilities for connecting a Z88 and a QL via a simple cable link. One was called QZ and thought it is not advertised now might be available from second-hand QL traders, the other was written by Phil Borman and is to be found in the Quanta Library and possi-

ble from the Z88 User Group too."

Z88 Eprom magazine, membership, and Spares Service (members only); also deals in Personal Organiser printer paper, ram upgrades (self-fit); Version 4 rom: Roy Woodward, Z88 Users' Club, PO Box 15, Belper, Derbyshire DE56 0XE. Answerphone/fax 0773 828707. (Club enquires with SAE please.)

Club software library: Ian Braby, The Software Library, Z88 Users Club, PO Box 210, Woking, Surrey GU21 1ZX. (All software enquiries to that address, with SAE.)

Z88s and supplies

Ranger Computers Ltd., Ranger House, 2 Meeting Lane, Duston, Northampton NN5 6JG. Tel. 0604 589200.

HiTek Marketing, 72 Longmoor Lane, Breaston, Derby DE72 3BB. Tel. 0332 873987.

Editor's notebook

There's been a bit of sniping about magazines, reviews and advertisers elsewhere in the QL press (who no doubt feel, as we do, that they must represent their readers' views). Leaving aside any slur on the people who write about the QL, and the question of whether any "expert" on the face of this earth could provide enough information to compensate entirely for everyone else's inexperience (in 20 years I have never yet found one. Computers. Origami. Car maintenance. Cookery. They all require some suffering for your art.) Leaving all that aside: Whereas it's true that some publications (like QL World) depend on some advertising, it is always possible to publish a magazine without any advertising at all. What it is really not possible to publish without - by definition - is readers.

Are any of us - magazine, publishers, advertisers - really going to mess you about that much? OK, we may be human, but think about it!

I'll be looking quite soon for a regular user to review the new DJC printing utility Sidewriter for QL World, at our usual rates. Do we have any volunteers?

Italy Show

Don't forget the 5th **Italian QL Meeting** - organised by Ergon Development and the QItaly club - in Reggio Emilia on Sunday 26 September 1993. Early information can be obtained from Davide Santachiara on +39 522 70409 or Eros Forenzi on +39 342 492323.

QL-I/O-Card For Control Applications

from Jurgen Falkenburg of JFC Computer Technik

Following encouragement by some customers last winter we decided to bring out a completely new QL product: the first professional and multifunctional input/output-board for the QL, the intelligent I/O card.

The **I/O-Card** is a standard expansion board for the QL system bus. It is free addressable inside the complete QL rom memory, and Gold Card compatible. There are almost no restrictions to its applications to allow any QL to communicate with its environment in terms of measurements and controls. With a high-speed A/D (8-bit, 2 microseconds) and D/A converter (8-bit, 1 microsecond) and 32 digital input/output lines, many connections are possible. With the included Basic toolkit, all I/O functions may be performed comfortably.

Co-processor

The extra add-on is the integrated 24-MHz 8051 co-processor. The I/O-Card I/O functions may be controlled from the QL itself (with the on-board toolkit) or from the co-processor. It is controlled with a single further command of the QL-toolkit, and has lots of pre-programmed functions. With many applications the QL remains completely free for system control and as a constantly-available user interface. You can even do word-processing or programming work, while the I/O-Card performs complicated controls.

Five of the digital I/O lines may be used for special applications: two counter/timers, a serial interface and external co-processor interrupt. A special register is used for control of and command transfer to the co-processor from the QL. The 8051 is installed with 32K rom and 32K ram. The dual-port ram may be accessed from the QL in pages of 8K. In this way, amounts of data up to 32K may be transferred easily between the processors. For special applications 8051 programs may be downloaded from the QL into the ram and called there. Very flexible and powerful I/O applications can be realised.

Those who are able to program the easy-to-learn 8051 assembler may perform solutions for special applications. The QL will remain free for system control and constant user input. After the straightforward testing of your 8051 code in ram, the software may be transferred in the free memory of the 8051 eeprom.

Applications

The range of applications for the I/O-Card is nearly unlimited: scanner, sound-sampler, voltmeter, measurement of any sensor signals such as temperature, brightness, pressure, humidity, distance, rotation, speed, etc., measurement amplifier, delay- and echo-generator, digital storage oscilloscope, counter, timer, free programmable

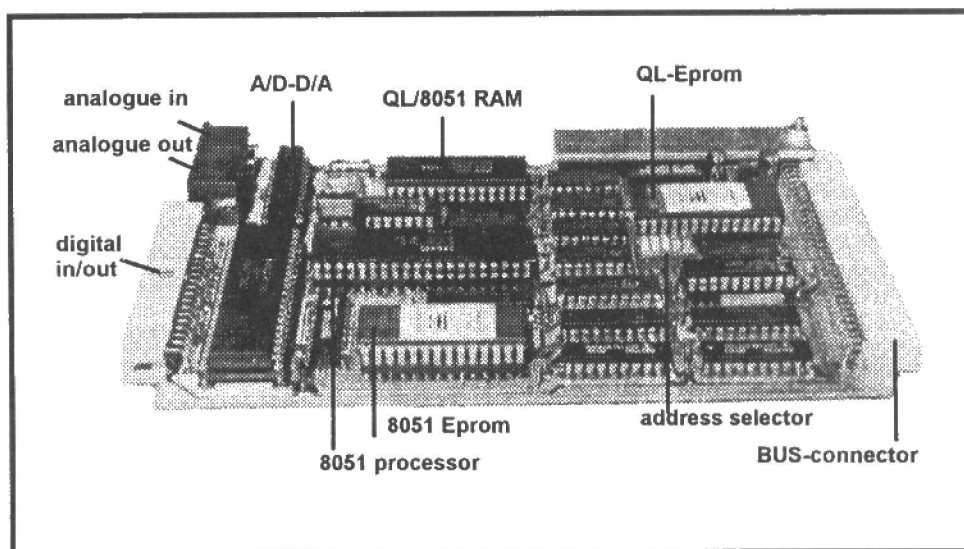
frequency generator (up to 1MHz), free programmable square generator (32-channel, 200kHz or 8-channel 1 MHz), logic analyser with up to 32K memory depth (32-channel 200kHz, 8-channel 1 MHz), interfaces with up to 32 signal lines like centronics, RS232, IEEE-488, SCSI, I2C-bus, Midi etc, control of up to 32 relays or 8 bipolar stepping motors, model train control, round-counter and speedometer for 16-lane racing car model, traffic-light control for a little town, and so on.

If you want to control the production line of a car manufacturer, for instance, no problem: beside ram expansion, floppy and hard disk, up to 16 I/O-Cards may be plugged into a standard QL if it is supplied with a special bus driver and power supply. With 128 stepping motors, many movements may be controlled. If this does not suffice, simply install a QL-network. Due to the multitasking abilities of the QL, even the software for such an application may be developed with modest effort.

With the circuit suggestions in the manual, individual peripherals (relay driver, motor driver, amplifier, etc.) can be assembled even by those who are not highly skilled in electronics. Ready to-use accessories will become available soon, as well as the proven peripherals of our A/D-product series.

Introductory price

As comparable boards for IBM- or industrial computers in industrial versions come at industrial prices (£1000 and upwards) we are



trying, with a presentation offer of about £160, to encourage a great number of users to look at modern computer control. The popularity of the first series of the I/O-Card will determine the quantity and price of possible following series.

As always in the past, with the I/O-Card we tried to develop an innovative product expanding the QL's horizons. We hope that the simple QL and I/O-Card handling will give an entry into computer-controlled measurement and control applications to the less experienced. For specialists, the I/O-Card will fill this gap in the QL peripherals product range for the first time.

Suppliers: JFC Computer Technik, Thanweg 36, D-75236 Ersingen. Tel. +49 7231 81058

UK distributor: W.N. Richard-son & Co, 18-21 Misbourne House, Chiltern Hill, Chalfont St. Peter, SL9 9UE. Tel. 0753 888866.

The I/O-Card will be available from July. The starting offer will be DM 399 (inc. VAT) all over the EC, plus postage of DM 20.

Hardy Hints

Perennial solutions to perennial problems. If you have any favourites, please send them in.

Seconds out

With reference to the Clock Quirk mentioned by Bryan Davies in Troubleshooter, November 1992: the QL clock starts at 1961 Jan 01 000000, and goes on until 2097 Feb 06 06:28:15, and you can SDATE any time between these limits. Calculations are done in seconds, starting DATES(0), ie Jan 1 1961, and the limit is the largest number (of seconds) that can be stored in 32 bits. The way in which very large numbers are stored in two's-complemented form is explained on page 4 of Andrews Pennell's Assembly Language Programming on the Sinclair QL.

The "quirk" occurs after 2029 Jan 19 03:14:07, where Date = 2147483647 (seconds). When DATE reaches a higher number than this, the Function returns the number of seconds - 2147483648. The following SuperBasic lines, instead of those given by Bryan, will show DATES and DATES(0) as the same up to 2090, and if you care to continue STEP 1 instead of 10, you can go up to the clock's limit, 2097 Feb 06 06:28:15. After that, you get back to DATES(0).

```
100 FOR z=1970 to 2020
    STEP 10
110 SDATE z,1,1,1,1,1
120 PRINT DATES,
    DATES(0),DATE
130 END FOR z
140 x=2147483648
150 FOR y=2030 TO 2090
    STEP 10
160 SDATE y,1,1,1,1,1
170 PRINT
    DATES,DATES(0),DATE-x,
    DATE-x
180 END FOR y
```

While not many people will be interested in dates up to 2097, many historians may be interested in dates before 1961. I came up against these limitations of the QL clock most recently in writing a program which prints out a tabular

calendar for any year. For anyone interested in dates outside the clock's range, I enclose a listing for a DOW FUNCTION, which returns the Day Of Week for any date as a number 0 to 6, where 0=Sunday. This is, of course, for our present (Gregorian) calendar, but it can easily be adapted to work for the Julian calendar as well.

```
100 DEFine FuNction
    dow(day,MONTH,year)
110 LOCAL x,z,m
120 x=0
130 IF year MOD
    4=0:leap=1 ELSE leap=0
140 IF year MOD 100=0
150 IF INT(year/100) MOD
    4=leap=0
160 END IF
170 z=year-1601
180 dif=z
190 z=z+INT(dif/100)
200 z=z-INT(dif/100)
210 z=z+INT(dif/400)
220 m=MONTH
230 SElect ON m
240 -2z=z+3
250 -3,1,1z=z+3:IF leapz=z+1
260 -4,7z=z+6:IF leapz=z+1
270 -5z=z+1:IF leapz=z+1
280 -6z=z+4:IF leapz=z+1
290 -8z=z+2:IF leapz=z+1
300 -9,12z=z+5:IF leapz=z+1
310 -10z=z+0:IF leapz=z+1
320 END SElect
330 z=z MOD 7
340 x=(x+z+day) MOD 7
350 RETurn x
360 END DEFine dow
370
```

**G Jackson, Whitby, N
Yorkshire**

Long Words

Regarding the "Clock Quirk" comments in Troubleshooter November 1992 and February 1993, in the QL the time (in seconds) is stored in the clock port register (PC_CLOCK) located at address \$18000 (D98304). This register is a Long Word (32 bits) and the time is stored as an un-signed Long Word Integer. This will allow a

maximum time (relative to 1961 Jan 01:00:00) of 2097 Feb 06:06:28:15. This equals (2 to the 32nd)-1, or 4,294,967,295 seconds. However, different portions of the date/time from 1961 Jan 01:00:00:00 to 2097 Feb 06:06:28:15.

The SuperBasic SDATE and simple DATES commands (without any numeric parameters) treat this number as an un-signed Long Word Integer of 32 bits. This will allow for the clock to be set using SDATE, or read using PRINT DATES, to any date/time from 1961 Jan 01:00:00:00 to 2097 Feb 06:06:28:15.

The PEEK.L command treats the value in PC_CLOCK as a signed Long Word Integer. In this case, bits 0 to 30 indicate the value, and bit 31 (the MSB) indicates the sign. When bit 31 is not set, the number is positive, and when bit 31 is set, it is negative. This gives a range of values from -2 to the 31st (2,147,483,648) to +2 to the 32nd-1 (2,147,483,647). Therefore, until 2029 Jan 19:03:14:07 PRINT PEEK.L(98304) will return an increasing positive floating point value. After this time, PRINT PEEK.L(98403) will return a diminishing negative value.

Although PC_CLOCK may be read with PEEK.L, no attempt to alter the value of its contents with the POKE.L command should be made. This will only zero the contents of the register. Only the correct commands should be used to alter the contents of PC_CLOCK.

The PRINT DATE command treats the value in PC_CLOCK as an un-signed 31-bit Long Word. It disregards bit 31 (the sign bit) and returns the value in the lower 31 bits. It will only respond to numbers in the range 0 to +(2 to the 31st -1). It completely disregards the sign bit (bit 31). Therefore after 2029 Jan 19:03:14:07, it will start again from 0 (1961 Jan 01:00:00:00).

So, provided the time is set correctly, PRINT DATES will return the correct time until 2097 Feb 06:06:28:15. But, PRINT DATES(0) will only return the correct date up to 2029 Jan 19:03:14:07. After this time, it will start returning the date from 1961 Jan 19:03:14:07 onwards.

The following SuperBasic listings illustrate the above points:

**P Hutley, Dewsbury,
West Yorkshire**

Listing one

```
1 OPEN#5:SCR_512X240A0X0:CLS#5
2 FOR X = 0 TO 10 STEP 1
3 SDATE 1961,1,1,0,0,X
4 PRINT#5,X,DATES,DATES(PEEK.L(98304)),
    PEEK.L(98304),DATES(0),DATE
5 END FOR X
```

Listing two

```
1 OPEN#5:SCR_512X240A0X0
2 FOR X = 0 TO 10 STEP 1
3 SDATE 2029,1,19,3,14,X
4 PRINT#5,X,DATES,DATES(PEEK.L(98304)),
    PEEK.L(98304),DATES(0),DATE
5 END FOR X
```

Listing three

```
1 OPEN#5:SCR_512X240A0X0
2 FOR X = 10 TO 20 STEP 1
3 SDATE 2097,2,6,28,X
4 PRINT#5,X,DATES,DATES(PEEK.L(98304)),
    PEEK.L(98304),DATES(0),DATE
5 END FOR X
```


Fleet Tactical Command

Version 2

Alan McTavish longs for a life afloat

INFORMATION

Program: Fleet Tactical Command Version 2

Supplier: DJC, 41 Bro Emrys, Tal-Y-Bont, Bangor, Gwynedd LL57 3YT. Tel. 0248 354023.

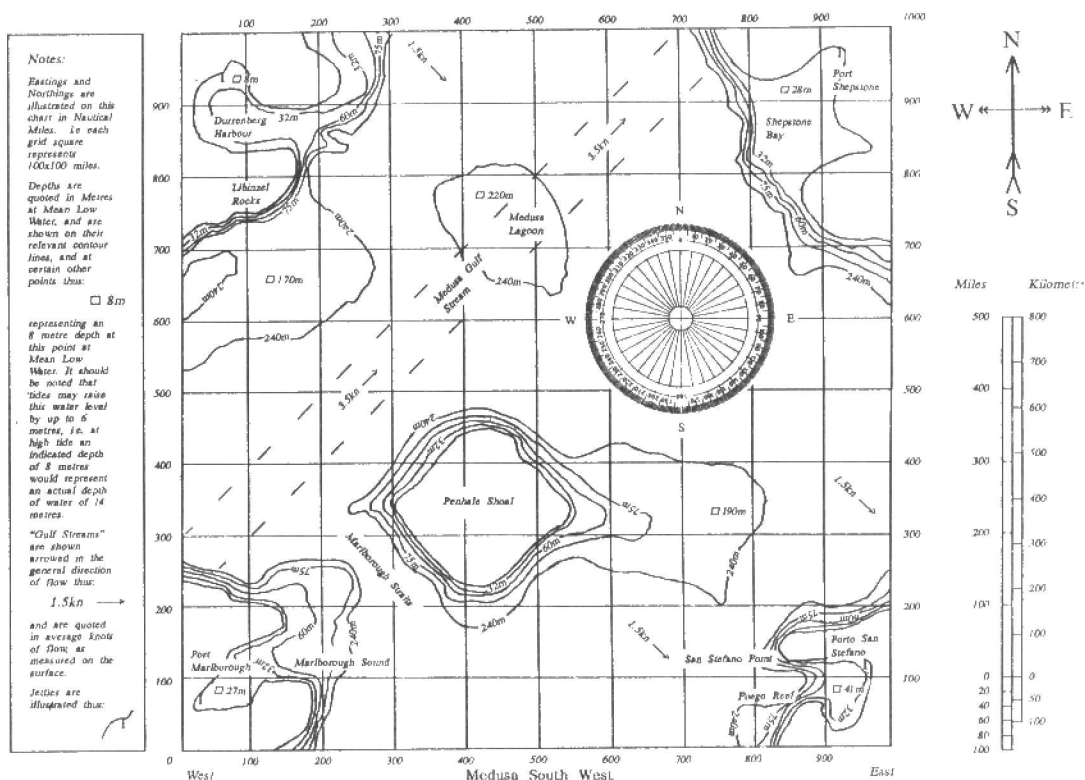
Prices: QL version now £39.95; PC version £49.95; PC/QL combined package £75.95 (serial link not included).

No doubt many QL World readers are already familiar with Fleet Tactical

Command. For those who are not, I will recap briefly (I should point out that it is impossible to do justice to this program in a short overview).

FTC is a real-time naval wargame simulator. The program can be used on one or two QLs linked via serial/network options. Each player commands a fleet of 16 assorted war and supply ships. Each ship has to be individually controlled by the player either being "on" it, or sending signals to it. You can move from ship to ship quite easily.

The scenarios set in a 1000x1000 mile area within which there are ports and natural hazards.



Bridge view

A lot of thought has obviously gone into the display which is divided into three distinct sections. At the top is a line of digital information such as echo sounder readouts, fuel, capability readouts, etc. Underneath this is the bridge window that allows a view from any angle of 180 degrees. If any of you own the enemy ships are within range they are displayed in a wire-frame 3D image (QL version) upon an animated sea.

Below the bridge window is

the command input console, tactical radar display, ship's IFF computer, and other useful instrumentation.

The makers of FTC, Di-Ren, have justly claimed to have made the program as realistic as possible. The ships, for instance, move in a manner consistent with their bulk and weight. The manuals include a section devoted to navigation for which purpose charts and geometry sets are provided. Tide flows, levels and natural hazards have to be taken into account when plotting courses.

Ships have all sorts of com-

partments and machinery fitted. The manuals contain a wealth of information including, for instance, a ship's wiring diagram. Machinery is of course liable to break down or can be damaged during an engagement. Compartments are likely to be affected by flooding and fires, problems that all require solving.

More than a game

FTC is stuffed with details and is, to the best of my knowledge, unique. I am not normally a

computer games player, confining that sort of activity to chess or card games, but FTC is something different. One big problem with the program is its addictive nature, something is always happening that requires your attention!

I first purchased FTC after its original review by Bryan Davies a couple of years ago. At that time program development was continuous and a number of upgrades were received from the developers, each providing new features and a more fascinating game.

Eventually version 2 became available. This upgrade contained several major improvements over version one. Di-Ren obviously made a conscious decision to remove some of the "frills" and concentrate on more useful features. Gone, for instance, were the nuclear missiles, and in came a much-needed towing option. The crew became part of the manage-

ment resource situation. One could take on crew at their home ports and even transfer crew from ship to ship while under way. Given the right circumstances, survivors could be rescued. The command input parser was improved to accept multiple commands and abbreviations.

Improvements

By far the greatest improvement was a general tightening up on the scenario. In version 1 you had to enforce a total exclusion zone by tracking and disabling the enemy fleet. This could be a somewhat tedious business. Version 2, however, forces opponents to use the same neutral ports thereby ensuring contact is made. Also, instead of being effectively locked out of the game for several hours (game time) for viola-

tion of neutral territory, you are simply bombarded by shore batteries!

A further major advance with version 2 was the introduction of a two-player option on one machine. This option is invoked upon command or an automatic timing system.

Fleet Tactical Command II was also the version that would be compatible with other machine versions.

In 1992, Di-Ren announced that the long-awaited PC version was ready to be shipped. I obtained this version as an upgrade. To have re-programmed FTC to run on PCs is something of an achievement. As with the QL, the program has been produced entirely in machine code. I remember Di-Ren stating in the early days of FTC that machine compatible versions would be produced, and I considered it something of a pipe dream! Hesitatingly, I connected the QL and PC with a

serial link lead and was amazed to find that the program runs across the two machines with no apparent problems.

PC version

The PC version of course has a significantly improved display. The ships are in-filled graphics, and there is also a "tilt" option that adds to the realism, especially in rough weather!

Finally, FTC was never very cheap, and when you receive the package you understand why. As I understand it, Di-Ren have recently reduced the QL version to £39.95. And that's a bargain! Take my word for it!

Here I go again. Five hours into the game, and I've put three ships aground, sunk two, have one under tow and (worse) another short of supplies. When will I learn? Never run out of coffee...

QUANTA

Independent QL/Thor Users Group

Worldwide Membership is by subscription only, and offers the following benefits:

- Monthly Newsletter – up to 40 pages
- Massive Software Library – mostly FREE!
- Free Helpline and Workshops
- Regional Sub-Groups. One near you?
- Advice on Software & Hardware problems
- Discounts from most major suppliers
- Subscription just £14 for UK members
- Overseas subscription £17

Barclaycard: Visa: Access: Mastercard

* Now in our NINTH successful year *

Further details from the Membership Secretary



Bill Newell
213 Manor Road
Benfleet
Essex. SS7 4JD
Tel (0268) 754 407



IF AN ADVERT IS WRONG, WHO PUTS IT RIGHT?

We do.

The Advertising Standards Authority ensures advertisements meet with the strict Code of Advertising Practice. So if you question an advertiser, they have to answer to us.

To find out more about the ASA, please write to Advertising Standards Authority, Dept. X, Brook House, Torrington Place, London WC1E 7HN.



This space is donated in the interests of high standards in advertisements.

VERY BASIC SUPERBASIC

*In this issue, Dilwyn Jones starts **SELECTing** what he'd like to write **ON**.*

We looped and made decisions in the last issue, now we shall look at the use of the **SElEct** statement to make more decisions.

SELECT

SELECT is a useful little keyword allows us to make decisions which depend on the value of a variable, or a number. Although in most cases we could use **IF...THEN** to make equivalent decisions, a **SELECT** statement is often more useful when there is a large number of choices which can be made. An example is worth a thousand words here, so study **listing one**.

It consists of three main parts. The beginning of the decision making process is marked by the "**SELECT ON a**" statement. The end of it all is marked with

an **END SELECT** statement, while there are various statements starting with '-' between them.

The program chooses a random number from 1 to 3. It then proceeds to translate the number into English words. The **RND** keyword simply chooses a number at random between the limits specified in brackets after it, in this case a whole number from 1 to 5 inclusive.

"a" variable

Firstly, the **SELECT ON** statement specifies that decisions will be made based on the value of a variable called "a". The decisions are placed on separate lines between the **SELECT ON** statement and **END SELECT** statements. **ON** can actually be omitted, so you only have to type "**SELECT a**", although using the full form is a little bit clearer and more meaningful. In fact, there is an abbreviation for the **SELECT** keyword as well - you only have to type the three letters **SEL** for it to be recognised,

Listing one

```
100 CLS
110 LET a = RND(1 TO 5)
120 PRINT 'a is ';
130 SElEct ON a
140   =1 : PRINT'One'
150   =2 : PRINT'Two'
160   =3 : PRINT'Three'
170   =REMAINDER : PRINT'Something else.'
180 END SElEct
```

though I normally type it in full to avoid the chance of errors.

The lines in between specify what action to take for each value which the variable takes. They all take the form "-number", where number is actually a number here, though it can also be a variable if you wish. It can also be a list, with values separated by commas, or a 'range' like in the **RND** statement, using the **TO** separator.

The statement

"-1 : PRINT'One'"

basically means "IF a=1 THEN PRINT'One'" as we learned last month, meaning that when it discovers that a has a value of one, it prints the word One on the screen. A fairly trivial little application, but it clearly shows how **SELECT** works.

You may have noticed that we

Listing two

```
100 INPUT'What is your age?';age
110 SElEct ON age
120   =0 TO 19
130     PRINT'A young QL user.'
140   =20 TO 29
150     PRINT'In your twenties.'
160   =30 TO 39
170     PRINT'Thirty something,'
180     PRINT'like the author!'
190   =REMAINDER
200     PRINT'Mature and experienced!'
210 END SElEct
```

Listing three

```
100 LET compliment = RND(1 TO 5)
110 SElEct ON compliment
120   =1 : PRINT'Hello, gorgeous.'
130   =2 : PRINT'You are looking smart today.'
140   =3 : PRINT'You must be the most intelligent'
150         PRINT'QL user I have ever met!'
160   =4 : PRINT'I wouldn't swap owners for the world.'
170   =5 : PRINT'Aren't you a super person!'
180 END SElEct
```

Listing four

```

100 CLS
110 REPEAT loop1
120   INPUT'Month number (1-12):';month
130   SELECT ON month=1 TO 12:EXIT loop1
140   PRINT'Oops, invalid entry!'
150 END REPEAT loop1
160 SELECT ON month
170   =4,6,9,11 : no_of_days = 30
180   =2 : no_of_days = 28
190   =REMAINDER : no_of_days = 31
200 END SELECT
210 REPEAT loop2
220   INPUT'Day number:';dno
230   SELECT ON dno=1 TO no_of_days:EXIT loop2
240   PRINT'Oops, invalid entry!'
250 END REPEAT loop2

```

only test for the numbers 1,2 and 3, whereas the random number could also be 4 or 5. In this case, we include a "REMAINDER" statement which executes if none of the other has caught the value. What this means is that if "a" does not have one of the other values (1,2,3), the command after REMAINDER is executed. In this case it simply prints "Something Else".

We could also have written that statement in these ways, to show how the lists and ranges work

-4,5

or

-4 TO 5

Lists and ranges can be combined into long sequences of values if you wish, though at first this may seem confusing. A statement such as this is quite legal, though in this example it is clearly of little use:

-1,2,3,4 TO 5,26,30 TO 40

Variables can be used too, though again of little use in this example. Variables tend to be less clear than the numbers and are not used as often.

100 SELECT ON a

```

110 -b:PRINT'Same as b'
120 -c:PRINT'Same as c'
130 END SELECT

```

PRINT is not the only command which can be used, all QL commands can be used, though some are not very useful or practical, such as NEW which would destroy the program!

Incidentally, if you use a "REMAINDER" clause, it should be right at the end of the list of "-" statements, to prevent it catching values not caught by other "-" statements which might be after it. In other words, it should be the last "-" statement before the END SELECT.

Action anywhere

In all examples so far, we have simply put the action statements or commands on the same line as the "-" statement, separated by a colon. This does not have to be the case, the action statements can be anywhere before the next "-" statement - an example of this is in **listing two**.

One small matter needs to be pointed out and this may be difficult to understand at this stage. When testing for the value of a variable, it actually means

'approximately equal to' not 'exactly equal to', to allow for slight inaccuracies in the use of decimal numbers. According to Jan Jones, the original designer of SuperBasic, in her book **'QL SuperBasic, The Definitive Handbook'** (available from Quanta) the test allows for small rounding off errors of up to the seventh decimal place (plus or minus 1E-7). You may be familiar with the description 'accurate to 7 decimal places' sometimes given to calculators. In practice, what it means is that "-6" might not catch the value 5.9, but it should be able to catch 6.000000001 easily enough. This is significant where you are testing for the result of a calculation, or trying to match the value of something like 1/3 which cannot be specified exactly.

Don't forget that you can test for decimal numbers as well as whole numbers, though on original QL ROM versions, you can only use floating point variables - you can't use integer variables (which only hold whole numbers and have names ending with the % symbol).

Range checking

The 'approximately equal to' rule does not apply to range checking. If you specify 4 TO 6, for example, it means just that, exactly equal to 4, exactly equal to 6, or somewhere in between.

There is also a short form or 'in-line' version of the SELECT statement which can all be written on one line of a program! In

this the value checks and the SELECT statement are all on the same line and there is no END SELECT statement. All parts are separated by a colon. You can include multiple statements, or multiple "-" clauses if you wish, although one is the norm.

```

SELECT ON a=1 TO
3:PRINT'That is OK!' SELECT ON
a=1:PRINT'One':-2:PRINT'Two'

```

That was the theory. Let us now look at a few simple and practical examples. **Listing three** shows a fun little application of the SELECT statements, a compliment generator to cheer up any fed up QL users! You could also change this to an insult generator (!), or for a bit of fun, into a buzzword generator which selects long words at random to form impressive sounding phrases which you can casually drop into conversations to sound impressive!

Listing four shows a more practical use. This shows how to enter a month and day number and use SELECT to help prevent errors and ensure you get the right number of days in each month, although it doesn't test for leap years, where there would be 29 days in February.

The routine works by using short form SELECTs to check the ranges of values allowed and long forms to specify the number of days in a given month number. As normally specified, January is month 1 and so on. There are other, better ways of writing such a routine, but it is an excellent example of the use of SELECT.

Listing five

```

100 CLS
110 LET a = RND(1 TO 3)
120 ON a GO TO 1000,2000,3000
130 STOP
1000 PRINT'One' : STOP
2000 PRINT'Two' : STOP
3000 PRINT'Three' : STOP

```

Listing six

```
100 CLS
110 LET a = RND(1 TO 3)
120 LET line_number = 140
130 ON a GO TO 1000,2000,3000
140 PRINT'Program finished.'
150 STOP
1000 PRINT'One' : GO TO line_number
2000 PRINT'Two' : GO TO line_number
3000 PRINT'Three' : GO TO line_number
```

GOTO

You may have wondered if, apart from ensuring that lines of a program are sorted in the correct order, the line numbers have any other significance. The answer is yes - it is possible to make the program jump from its present line to another in a different part of the program. To do this, we use the GOTO statement, which is followed by a line number. In the following example, line 110 is never executed when the program is started from the beginning:

```
100 GOTO 120
110 PRINT'Line 110'
120 PRINT'Line 120'
```

It can also be made to form a 'loop', rather like REPEAT:

```
100 PRINT'Hello!';
110 GOTO 100
```

In fact, you'll notice that the computer enters a GOTO command as GO TO (with a space between GO and TO). You do not have to type it in like that, it will be quite happy for it to be entered as one word.

The GOTO statement has little practical use in QL SuperBasic, since there are other commands which can be used to do

the same things. Its disadvantage is that by specifying line numbers it makes it difficult to follow a program, and difficult to figure out where blocks of code start and end. It is, however, very useful when trying to convert programs written for other computers (Basic programs listed in a magazine or book) since these make more use of the command than the QL does.

ON...GOTO...

There is another way in which you can make decisions about what to do subject to the value of a number or variable. This one is shorter and some times more convenient than SELECT, but suffers from the problems with line number references discussed under GOTO above. This is also included in Super Basic because it appears in other versions of Basic.

The command takes the form

```
ON x GOTO 1000,2000,3000
```

where x is the variable being tested and the numbers after GOTO are line numbers to choose. These signify the start of blocks of code to handle the result. It does not appear obvious how the choice is made, but it is quite simple. If the value of x is 1, the first line number is cho-

sen. If the value of x is 2, the second is chosen and so on. If the value is too low or too high for the amount of line numbers you've given, the computer gives an error report ('Out of range') and the program stops.

Listing five is a version of listing one, rewritten to use ON...GOTO...

Don't forget that the use of GOTO is frowned upon in QL SuperBasic, though there's nothing to stop a consenting adult using it in private with his/her QL!

Getting lost!

When using GOTO or ON...GOTO to make a program jump to another part of the program, it is possible that it might get lost and not know where to come back to after it has finished the new block. Programs are often written in sections like this, especially where two parts of a program use the same lines of code. It allows you to avoid duplication and repetition - a part of a program can be written as one chunk at the end and called with a GOTO from one point and from another point, though you have a problem trying to remember where to go back to. One solution is to use a variable to remember it. Here, I will be using a REMark statement to mark out bits of code as

examples. A REMark statement is just that, a remark or comment to include information for you to read. It is ignored by the computer. The example is in **listing six**.

Here, the variable line_number holds the number of the line to go back to after executing the block at the end of the program. This works, but is not very elegant (in fact it is downright horrible).

SuperBasic provides a far better solution to all this. It has a command which allows you to jump to another section of code, remembering automatically where it came from. At the end of a section of code is a RETURN command which tells the computer to go back to just after where the last call was. This block of code at the end of the program is called a subroutine, because it is not the main routine. The command to call it is not surprisingly GOSUB (which means go to subroutine). The GOSUB command is followed by a line number. When encountered, the program jumps to that line number and continues until it reaches the RETURN statement, then jumps back to where it left off. See the example in listing seven.

SuperBasic provides structures which are far more clever than this, doing away with the need for line numbers altogether. These are called procedures and functions and will be discussed in the next instalment.

Listing seven

```
100 CLS
110 LET a = RND(1 TO 3)
120 ON a GO SUB 1000,2000,3000
130 PRINT'Program finished.'
140 STOP
1000 PRINT'One' : RETURN
2000 PRINT'Two' : RETURN
3000 PRINT'Three': RETURN
```


A Fair Xchange

Simon Goodwin reports on the long-awaited release of Xchange for the QL

This year Psion Plc gave permission for the West Midlands QL User Group to distribute the full Psion **Xchange** suite to QL users. This package is potentially interesting to all QL users, as it may be freely copied and is a major update on the Psion software bundled with each QL. You need at least 384K ram and one disk drive to run the suite - the more the better.

Part-Xchange

This article explains the extra features in this version of Xchange, and discusses useful files which have been added to the PD release. Pirate copies of Xchange have circulated for several years, hacked from the Thor version. These should not be confused with the PD release, which has a cursor in the top left corner, so it will run with EXEC and multi-task with Basic, and comes with a full set of support files including printer drivers and tutorials.

All the versions hacked prior to permissions being given by Psion that I have seen contain errors; one, styled as 'Turbo Xchange', is bloated by needless extra code and will only work in conjunction with Qram or Qpac. Another hacked copy has 64 bytes missing from the start, so it shows no task name and you cannot control its memory usage. The new West Midlands version is 183718 bytes long, and comes on a 720K disk full but for three sectors. Accept no substitute.

Xchange story

Psion Xchange was born in 1983 as a suite of four business programs for the Sinclair QL computer. The first versions were programmed in C and tested on a Vax mini-computer; they barely fitted the 128K QL memory. Subsequently Psion updated the programs to 'version 2', re-coding them in assembly language to boost speed and save memory.

Xchange has four parts, familiar to QL users: **Quill**, **Abacus**, **Archive** and **Easel**. Sinclair supplied these as four separate programs, but the full Xchange suite combines them into a single task, about 180K long, with many extra features and improved communication between the parts.

Once QL Xchange was complete Psion converted the code to run on ICL's One Per Desk office workstation. This has the same 128K ram as a standard QL, and much of the same hardware, but uses a bigger rom which holds the entire Xchange package and built-in software added by ICL. The One Per Desk can run SuperBasic programs, but the interpreter must be loaded from microdrive cartridge before use.

For years rumours of a rom version of Xchange for the QL circulated, but it never arrived. Psion went on to convert the suite for IBM PC compatibles, launching the full Xchange at a price to match the entire QL system, and PC-Four, a cut-down version comprising four separate programs rather than the integrated set.

In 1986 the Thor consortium started work on a follow-up for the QL, and paid £30,000 for a Thor-specific version of Xchange, combining the four QL packages into one task and adding features from the One Per Desk version. In the event the conversion was much more work than expected, and went way over budget.

The Thor consortium tried to obtain further updates, but were unable to find the necessary finance. Thor version 3.90 was the last official release of Xchange, although a few jolly japesters have patched the version numbers of copies to make them seem more up to date.

On 23rd February 1993 Psion Plc generously gave permission for the distribution of QL-specific Xchange software freely (while retaining their rights), for the Sinclair QL and its derivatives plus the Thor XVI, Amiga and Atari running Qdos. The PC versions (Xchange and PC Four) are not affected by this decision, and must not be copied except under commercial licence from Psion.

A few days later former Dansoft boss Helmuth Stuvén contacted **Sinclair QL World** to announce that Thor bundled software was also freely distributable. This meant that QDUMP and other Dansoft accessories could be included in the new release.

Getting started

If RAM1_ is available you can start the Xchange task directly with an EXEC, EXECUTE or EX command. Xchange requires

about 500K including for default 300K data-area, and also uses space on RAM1_. You can reduce the data-area available for Xchange tasks by adding a parameter (eg EW "XCHANGE";90" for 90K) in an EW, QW or EXECUTE_W Toolkit command. There's no need for a 'grabber' to stop the task taking all your memory.

If you have no ram drive you can simply put the Xchange disk in drive 1, then load Xchange with the command LRUN FLP1_BOOT. This sets up a 50K ram disk; edit the SuperBasic if your machine has a built-in ram drive, or consult INTRO_DOC to find out how to change the size.

The main Xchange menu is new to QL users, and lets you start up to eight Psion tasks from the set of Quill, Archive, Abacus and Easel. It also incorporates options like BACKUP which used to occupy the files menu in each task.

You swap between tasks by pressing F6 to return to the main menu, then selecting the next task from a menu. Only one task runs at a time, but 'background' printing lets you print one file as you work with others. In general Xchange is faster than the QL tasks as David Oliver patched the code to remove redundant Qdos calls.

The eponymous XCHANGE option lets you move data between tasks, automatically swapping screens and issuing EXPORT and IMPORT commands, but this will not EXPORT from Quill, even though that now boasts an EXPORT option. More flexibly you can program arbitrary

command sequences with the TSL Task Sequencing Language, which works like 'macros' in other languages.

Task sequencing

TSL programs consist of a sequence of key presses, optionally interspersed with commands and line-labels. When you run the program, the commands and labels are filtered out, and the remaining characters are interpreted as though you were typing them at the keyboard.

TSL programs are text files which can be generated with any Ascii text editor, including Devpac, ED, Editor, Spy and the Xchange version of Quill, which can export Ascii text. Some QL Quill fans use a special printer driver to get the same effect. If you use Xchange Quill the easiest way is to press F3 O F for the files menu, then E to Export text, followed by the file name. The extension .TSL must be supplied, or Quill uses the default .EXP, which will not be recognised as a TSL file.

Spaces appearing in the program may be significant. Except within a TSL command, any space in a program is interpreted as though you had pressed the space bar. Other key-presses, like function keys, arrow

keys and Enter, are especially difficult to represent in a text file, so they have symbolic names which can be used instead of the character itself. These symbols use short mnemonics enclosed in square brackets. The **TSL table** shows valid symbols and their meanings.

The standard QL keyboard does not include F6 to F10, but the equivalent codes are generated by keying Shift with F1 to F5, so [f10] corresponds to Shift-F5, which re-draws the entire screen. The sequence [sf5] represents Shift-F5, and has the same meaning. Thor Argos, Amiga Qdos and many extended QL keyboards support the extra keys F6 to F10, and these come in handy when using Xchange - for instance, F6 calls up the main Xchange task control menu from any of the four packages, and F9 toggles between insert and over-write mode, like Shift-F4 in QL Quill.

TSL commands

You can include special TSL commands in a file as well as key-presses. Each command consists of an ampersand (&) followed by a distinguishing character. It must start at the beginning of a new line. Some

TSL COMMAND SUMMARY

&=	If the specified character matches the one pressed in response to the most recent &g, jump to the label specified. The test is case-dependent. To jump to @FUN if the last response to &g was ENTER, use: &= [cr] FUN
&c	Comment; the rest of the line is ignored.
&d	Display text to the user (see &p).
&e	End of program - stops TSL.
&g	Get character - waits until a key is pressed. The key press can later be tested with &=.
&i	INPUT; waits for text to be typed. After ENTER the text is passed to the task.
&j	Jump to the program line marked by the following label. To jump to the label @loop use: &j loop
&p 0	Disables TSL messages at the top of the screen, so Xchange prompts appear as normal.
&p 1	Permits TSL messages. This is the default.
&w	Wait for this number of tenths of a second. To wait for six seconds use: &w 60
&x	Wait for a specified key to be pressed; e.g. &x A waits for the capital A key to be pressed.
&y	Wait for a specified key to be pressed; ignore case.

XCHANGE

of the commands accept text or numeric parameters. TSL supports eleven commands, also listed in the table.

Lines may be labelled with text so that commands can refer to them by name. Labels names start with an at (@) character and must be the only thing on the line. You should not include the at sign in commands that refer to the label, such as &J, similar to Basic's

GO TO, or &=, which resembles an IF test.

The special meanings assigned to square brackets, @ and & signs, make it difficult to enter them literally in TSL, so Psion use ^ as an 'escape' character, rather like \ in C or Unix. If you want TSL to take one of these characters literally, add a ^ before it, so &= ^@ HOME sends TSL to the line labelled HOME if the last key pressed was @, and &x ^^ waits for the ^ key-press.

To start a TSL tutorial, press F6 (or Shift-F1) to return from the current application if the Xchange menu is not yet displayed, then press F3 and T, then enter the TSL file name.

Documents

The new documentation consists of a two page introduction and three pages of TSL details, plus 26 demonstration .TSL files which show the extra features of Xchange in action. These tutorials range in size from a hundred bytes to 30K or more, and have been collated from several sources including CST, Dansoft and Chas Dillon, and adapted to link from

TABLE - Xchange TSL Key Codes

Code	Equivalent key	Code	Equivalent key
[cr]	Carriage Return (ENTER)	[lt]	left Arrow
[esc]	Escape key (ESC)	[rt]	right Arrow
[sp]	Space (in TSL commands)	[up]	up Arrow
[f1]	Function key 1	[dn]	down Arrow
[f2]	Function key 2	[bpr]	SHIFT up
[f3]	Function key 3	[epr]	SHIFT down
[f4]	Function key 4	[wlt]	SHIFT left
[f5]	Function key 5	[wrt]	SHIFT right
[f6]	Function key 6	[bln]	ALT left
[f7]	Function key 7	[eln]	ALT right
[f8]	Function key 8	[dlt]	CTRL left
[f9]	Function key 9	[drt]	CTRL right
[f10]	Function key 10	[dwlt]	SHIFT CTRL left
[sf1]	SHIFT F1	[dwrt]	SHIFT CTRL right
[sf2]	SHIFT F2	[dbln]	SHIFT ALT left
[sf3]	SHIFT F3	[deln]	SHIFT ALT right
[sf4]	SHIFT F4	[tab]	TAB
[sf5]	SHIFT F5	[btap]	SHIFT TAB

one to another, starting with BEGIN_TSL or MENU_TSL

The original THOR 1 README.DOC is included on the disk, along with the programs that it mentions that work on the QL. The package also includes about 100K of context-sensitive help files. If in doubt at any time, press F1 for help from Xchange itself.

The Thor's PC-style keyboard includes HOME and END keys which are intended to move the cursor directly to the start or end of the line. Xchange recognises these, and you can get the same effect on a normal QL keyboard with Alt-left and Alt-right.

The Xchange Quill Glossary is similar to Toolkit II's ALTKEY feature, or the key-define feature in Turbo-Quill (which is not compatible with Xchange). It lets you assign a sequence of characters to any alphabetic key, and call them up with the glossary key F5, unused in QL Quill.

To teach Xchange a Glossary key, press F5 twice, then the letter you want to assign, followed by Enter. From that point onwards until you press F5 again, Quill records all the keys you press, up to a maximum of 250 key-strokes. Once a glossary is set up you can call up the sequence by pressing F5 and the chosen letter.

Psion suggest that you could

use the glossary to save typing common phrases, like 'yours sincerely' or your own address. Current glossary definitions are saved when you quit Xchange and re-loaded when you start again from the same disk.

File types

The new EXTRACT option on Quill's Files II menu lets you write part of a file to disk in Quill's .DOC format, with formatting intact. You can then use MERGE to load it elsewhere. This is quicker than COPY even if you are just moving the block from one end of a file to the other, as you can use GO TO or other commands to move once the block has been extracted. By default the block is stored in BUFFER.DOC on the default drive, but you can specify an alternative name or device - I normally use RAM1.

Quill documents have two main parts - the text, and the paragraph table which keeps track of the format and grows periodically as the file expands. Quill generates a new paragraph table for extracted blocks, so you can often reduce the size of a .DOC file by selecting it all as a block, and extracting it; this eliminates space which may have been added to the paragraph table but not yet used. I use this tech-

nique to ensure that DIY Toolkit text files fit on 128K QLs, which limit Quill to documents of around 10240 bytes.

Ascii export is a long-awaited feature added to the FILES menu in Xchange Quill. EXPORT writes the entire file to a drive, removing control codes. Unlike SAVE it does not forget the cursor position, so you can carry on from just where you left off. MAIL is another new FILES option. It generates personalised form letters by merging a template document with data from the keyboard or a .DOC or .EXP file.

Better names

By default Xchange uses the same eight-character file names as QL packages, adding a three-character PC-style extension to distinguish between file types, but one of the nice Thor extensions means you can specify a full file path, allowing direct access to sub-directories or files grouped by prefix.

You are not restricted to the three-character extensions, although you may avoid confusion if you stick with them. For instance F3 S flp1 .DIY_TEST generates a file called "DIY_TEST"; you have to specify flp1 .DIY_TEST.DOC to create

TEST.DOC in sub-directory DIY. The default devices are derived from the PROG_USE and DATA_USE names set up by most disk systems, but you can change these with the SET option from the Xchange F6 menu.

The first default is used to find data files, unless an explicit device is named. Curiously, it is also used to find printer files such as GPRINT_PRT and XCHANGE.DAT. The other default tells Xchange where to look for help files. These are extended versions of the help files used in the QL bundle, and include a new file, XCHANGE.HOB, which explains new commands in the Supervisory F6 menu.

Abacus

The Abacus spreadsheet recognises the same functions as the QL version. The FILES menu includes a new 'transfer' option to save or load the spreadsheet as an Ascii DIF file or in Psion's own binary .ABT format. There is no FORMAT option here or elsewhere, but you can Control C out to Basic any time you need to format a disk or cartridge.

The new TITLES command fixes outer rows to the border, so beyond the cursor scrolls while the headings stay put. The GRID menu gains many new options. PROTECT and UNPROTECT let you guard cells against overtyping over formula amendment. SECURITY protects spreadsheets with an eight character password. REPEAT re-calculates the number of times you indicate while GOALSEEK re-calculates until the supplied formula evaluates to zero.

Easel

Easel now supports 3D graphs, changing the viewing angle automatically so you can always see the height of all the bars. It gains some new functions: RAD and DEG work like their Abacus namesakes, while

HELP press F1	XCHANGE TASK CONTROL		COMMANDS press F3
PROMPTS press F2	Use space to select the task Press ENTER to start the task		XCHANGE press F6
TASK TABLE		MEMORY USED	DEVICES
QUILL - Xchange		8 Kbytes	default flp1_
QUILL - Action		4 Kbytes	help nl_flp1_psion_help_
ABACUS - Eric		6 Kbytes	
ARCHIVE - Simon		5 Kbytes	
EASEL - DIYlook		4 Kbytes	
ABACUS - NEW TASK			
QUILL - NEW TASK			PRINTER STATUS
ARCHIVE - NEW TASK			not printing
EASEL - NEW TASK			
exec flp1_HPDUIMP_TASK;"PRT"			

A Fair Xchange

CELL and CELLMAX return the indices of the 'cells' that hold columnar data.

Archive

Archive gains the LOAD USR command added to the Archdev development version. This loads position-independent machine code which can subsequently be called with the USR function. This takes two parameters. The first, a long word, is passed in and out of the code via register D0. The second must be a string variable. The address of a string, prefixed by a length byte, ends up in register A0. The code can modify the text but must not change the length.

The file must start with the characters "PMCO", followed by a word value - the length of the subsequent code, in bytes. About 1K of stack space is available and all the registers may be used. The code should end with an RTS instruction.

Printing

Psion's slow and memory-hungry INSTALL.BAS has been replaced with a compiled version, but otherwise works just like the QL equivalent. QL PRINTER.DAT files work as soon as you rename them to XCHANGE.DAT.

A new program, CHOOSEPRINTER.BAS, lets you pick your printer definition from a range of pre-set files for SER and PAR devices. These support about a dozen old matrix and daisywheel printers, but few recent models apart from Epson-compatibles. Of course you can edit the existing files if you find something that is almost but not quite what you need.

I have added a simple DeskJet definition for the most commonly-used Quill control codes, but this does not currently support superscripts and subscripts. If this bothers you, add the sequences tabulated in **Open Channel** in issue 5 (May) earlier this year.

Extension files

The public domain ramdisk from Amiga Qdos appears on the disk, in case would-be users need a way to set up RAM1_, which is required for temporary files as Xchange runs. XCOPY_BAS is a fast copier for the Xchange task file, which uses a temporary buffer in memory to save and load the entire file in one go. You need to install SuperToolkit 2, or load Mark J Swift's PD Toolkit (LRUN FLP1_PDTK_BOOT) to use XCOPY_BAS.

QDUMP_INSTALL.BAS has been altered to work on Sinclair roms, Amiga and Atari emulators, and Minerva (in one or two screen mode), by adding a SuperBasic SYS_VARS function to emulate the Thor rom equivalent. You may delete those lines from the end of the listing if you habitually use Argos or Turbo Toolkit 3, which implement SYS_VARS more flexibly as a resident function.

Limitations

There are still a few bugs in Xchange that were fixed in very late releases of the separate QL tasks. In particular, the index used by Archive is limited to 32K, as on QL versions before 2.38. The Archive program token format is the same as that used by Runtime Archive and ArchDev, and different from that of QL Archive version 2.

You can load Ascii program files with the extension _PRG, regardless of their source, but tokenised files with the extension _PRO may need to be converted before they will load. The easiest way to do this is to load them with your original copy of Archive and save them out as _PRG files.

Quill sometimes gets the cursor position wrong while editing, as was common before QL version 2.35. If this happens to you, press Shift-F5 (or F10 on extended keyboards) to redraw the screen and fix the cursor.

David Moseley and Rich Mellor of West Midlands Quanta report a bug in the Xchange version of Abacus, but this only affects late Minerva roms, like their version 1.93. When they try to Amend an Echoed formula the QL crashes in what Tony Tebby once memorably described as 'fart and pyjamas' mode, with a striped screen and random noise from the speaker and microdrives. The fault does not appear on a JM machine with Gold Card, Amiga JS, or Phil Spink's early Minerva 1.64, so this seems to indicate a new Minerva bug.

Psion have always had trouble coping when memory runs short, and this is true for Xchange, despite many fixes. The suite can get stuck if it runs out of memory on RAM1_. To avoid trouble, make sure there's plenty of room for temporary files on RAM1_ before you start. Xchange generates a small file XCHANGE.TMP for task details, and one temporary file - similar to DEF.TMP on microdrive - for every task in use. They disappear when you Quit from Xchange.

Obtaining Xchange

The new release of Xchange is available free to members of the West Midlands QL User Group, a Quanta sub-group which meets on the first and third Monday of each month at the Holloway pub, just off Birmingham's Inner Ring Road. Bring a blank 3.5-inch disk to any meeting to get your copy. Non-members should join us! Or send an empty 720K formatted disk to the secretary with return postage and packing and a cheque or postal order for two pounds, made out to **QL User Group (WM). Write to Mike Bedford White at 16 Westfield Road, Birmingham, B27 7TL, UK.** Xchange is also available from other Quanta groups, and Qdos PD suppliers like Qubbesoft and SJPD.

Payment Power

Peter Tomlin's QL keeps track of his household electricity costs.

We wondered why our electricity consumption varied so

much. It was not just at weekends when everyone was at home. I used my *Abacus* to set out daily meter readings, but I soon wanted more data to hand, and it became slow and laborious shifting around the spread-sheet all the time to get what I wanted.

So gradually I developed this program, which has proved very useful and with which we have been able to reduce our bills.

Simple SuperBasic

The SuperBasic program does not require much explaining. Some of us do not have access to extra toolkits with facilities to **RENAME**, and **OPEN**, **OVERWRITE** files, so things have been kept as simple as possible, and all on one disk. *Mov* can of course be substituted for *flp*.

When starting to use the program for the very first time, there is another program called "Starter", which initiates files and can also be used to clear off old data and begin again when too much information has accumulated.

At the start of both programs one has the option of amending Unit Price, and Standing Charge if they are different. Should you need to amend them in the middle of a

Examples of Screens

1

```

Last reading was 30753
What is present reading ? 30787
Enter time & date (hh:mm Dt Mnn Day)
e.g. 09:00 09 Jul Thu
->
    
```

5

Cost so far :

```

This reading -> 31271 Units used -> 271
Standing Charge £12.22 (units @ 7.51p)

Last reading -> 30998 Units used then
(on 16 Sep) -> 610

Cost so far is -> £32.72 = £20.5 + St.Chge £12.22
    
```

2

Commands

To calculate cost:	Enter
- at time of Official Reading:-	Met
- at any time:-	Now
- of nightly usage:-	Nu
Just to see previous usage:-	Seelog
Just to see last 2 bills:-	Look

3

	Last Bill	Jun Bill
Reading was	30998	30388
Units used were	610	669
Cost of Units	£45.81	£49.97
Standing Charge	£12.22	£12.16
Bill total	£58.03	£62.13
	as at 16 Sep	as at 10 Jun

6

Last reading	this reading	Units used	Cost of units used	Time & date of reading
30719	30733	14	£1.05	09:00 28 Jul Tue
30733	30746	13	£.97	09:00 30 Jul Thu
30746	30753	7	£.52	09:00 31 Jul Fri
30753	30787	34	£2.55	09:00 05 Aug Wed
30787	30793	6	£.45	09:00 06 Aug Thu
30793	30800	7	£.52	09:00 07 Aug Fri
30800	30822	22	£1.65	09:00 10 Aug Mon
30822	30848	26	£1.95	09:00 14 Aug Fri
30848	30880	32	£2.4	09:00 19 Aug Wed
30880	30899	19	£1.42	09:00 22 Aug Sat
30899	30935	36	£2.7	13:08 27 Aug Thu

4

```

Enter meter reading from this bill 29719
Now reading from bill before that 28801

From this bill
enter time & date (hh:mm Dt Mnn Day) - guess time
e.g. 09:00 09 Jul Thu
->
    
```

7

Official reading today, 16 Sep

```

30998

Standing Charge £12.22 (units @ 7.51p)

Last reading -> 30388 Units used -> 610
Last time -> 669
- on 10 Jun

This bill will be -> £58.03 Last bill -> £62.13
    
```

1. Input Screen for "Nu" Procedure.
2. Menu Screen.
3. Procedure "Look" output Screen.
4. Starter Program input Screen.
5. Procedure "Now" output Screen.
6. "Nu" output & "Seelog" Screen.
7. "Met" output Screen.

session the facility can be accessed again by RUNNING the program again. In the very beginning the facility would have to be used in the Starter program, and also in the Main program. But all this is only necessary if the Unit Price and Standing Charge have been changed by the Gas or Electricity suppliers.

The Menu

Next, there is the list of commands, or the Menu.

Procedure "Met", when used, will permanently update the record and let you know what the bill is, after the meter reader has been and before the bill arrives in the post. Whether or not you feel that you need this information straight away, it will be needed eventually to update the program.

The procedure "Now" only reads the record to tell you how much bill you have run up so far. Nothing is permanently altered by this.

"Nu" is the procedure I started with. It updates a continuous record of all your periodic readings with the time and date alongside, so that readings can be related to conditions on that day. After breakfast was the best time for us to take a reading, when only the fridge and the freezer would be in ongoing use to this evening.

The running record can be viewed at any time without causing alterations by using the procedure "Seelog".

The procedure "Look" will show both bills from the last two official quarterly readings.

The groups of figures printed in the oddly-numbered line 203 are simply to assist in spacing out the following line, and can be eliminated if desired.

Program "Starter"

```

100 MODE 4
110 WINDOW #1,462,206,26,10
120 WINDOW #2,462,206,26,10
130 WINDOW #0,462,40,26,216
140 PAPER #1,0:PAPER #2,0:CLS:INK 4:CSIZE 0,0:INK #0,7
150 UChge = 7.51:Schge = 1222 :REMark >>> all in pence! <<<
160 PRINT"\\\\" Based on: Units"!UChge;"p & Standing charge
    £";Schge/100;".
170 PRINT" -----      ***      *****"
180 PRINT"                                if correct enter OK"
190 PRINT"                                *****"
200 PRINT"                                if different enter No"
210 PRINT"                                *****"
220 REMark *****
230 DEFine PROCEDURE No
240 CLS
250 INPUT"\\\\" Enter unit price IN PENCE:- "!UChge
260 INPUT " Enter standing charge IN PENCE:- "!Schge
270 CLS
280 GO TO 160
290 END DEFine No
300 REMark *****
310 DEFine PROCEDURE OK
320 CLS
330 INPUT"\\\\" Enter meter reading from this bill"!a
340 INPUT " Now reading from bill before that"!b
350 c = a - b
360 d = INT(UChge * c)/100
370 PRINT" From this bill"
380 PRINT" enter time & date (hh:mm Dt Mon Day) - guess time if
    uncertain."
390 PRINT"                                e.g. 09:00 09 Jul Thu"
400 INPUT "                                --> "time$
410 DELETE flpl_test_file
420 DELETE flpl_newtest_file
430 DELETE flpl_meter_file
440 OPEN_NEW #5, flpl_test_file
450 OPEN_NEW #6, flpl_meter_file
460 PRINT #5, b\ac\d\tim$
470 PRINT #6, a\c\d\tim$(7 TO 12)\Schge/100
480 CLOSE #6
490 CLOSE #5
500 LRUN flpl_Bill
510 END DEFine OK

```

Main Program "Bill"

```

100 MODE 4
110 WINDOW #1,462,206,26,10
120 WINDOW #2,462,206,26,10
130 WINDOW #0,462,40,26,216
140 PAPER #1,0:PAPER #2,0:CLS:INK 4:CSIZE 0,0:INK #0,7
150 UChge = 7.51:Schge = 1222 :REMark >>> all in pence! <<<
160 PRINT"\\\\" Main Program"
170 PRINT" *****"
180 PRINT" Based on: Units"!UChge;"p & Standing charge £;
    Schge/100;".
190 PRINT" -----      ***      *****"
200 PRINT"                                if correct enter OK"
210 PRINT"                                *****"
220 PRINT"                                if different enter No"
230 PRINT"                                *****"

```



```

240 REMark *****
250 DEFINE PROCEDURE No
260 CLS
270 INPUT"Enter unit price IN PENCE:- " !UCHge
280 INPUT " Enter standing charge IN PENCE:- " !SCHge
290 CLS
300 PRINT:PRINT:PRINT:PRINT
310 GO TO 180
320 END DEFINE No
330 REMark *****
340 DEFINE PROCEDURE display
350 CLS:CLS #0
360 OPEN #6,scr_65x16a235x175:PAPER #6,7:CLS #6:INK #6,0
370 BORDER #6,2,2,2
380 OPEN #7,scr_65x16a235x138:PAPER #7,0:CLS #7:INK #7,4
390 BORDER #7,2,2,2
400 OPEN #8,scr_65x16a235x66:PAPER #8,0:CLS #8:INK #8,4
410 BORDER #8,2,2,4
420 END DEFINE display
430 REMark *****
440 DEFINE PROCEDURE DK
450 CLS:CLS #0
460 CSIZE 0,1:AT #1,2,11:PRINT"Commands"
470 CSIZE 0,0:AT #1,4,11:PRINT"-----"
480 AT #1,7,11:PRINT "To calculate cost: Enter"
490 AT #1,8,48:PRINT "-----"
500 AT #1,9,16:PRINT "- at time of Official Readings:- Met"
510 AT #1,11,16:PRINT "- at any time:- Now"
520 AT #1,13,16:PRINT "- of nightly useage:- Nu"
530 AT #1,15,16:PRINT "Just to see previous useage:- See log"
540 AT #1,17,16:PRINT "Just to see last 2 bills :- Look"
550 END DEFINE DK
560 REMark *****
570 DEFINE PROCEDURE cue
580 PRINT #0," Enter OK for Menu"
590 END DEFINE cue
600 REMark *****
610 DEFINE PROCEDURE Met
620 LOCAL r,s,t,u,v#m,x,y#z
630 OPEN_IN #5, flpl_meter_file
640 CLS:CLS #0
650 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
660 INPUT" What is present reading ? " !r
670 PRINT" Enter date (Dt Mmm)"
680 PRINT" e.g. 09 Jul"
690 INPUT" --> " !ys
700 CLS
710 display
720 PRINT #8," " !r
730 PRINT
740 PRINT
750 PRINT
760 INPUT #5, s
770 INPUT #5, t
780 INPUT #5, u
790 INPUT #5, v#
800 INPUT #5, z
810 CSIZE 0,1:AT #1,1,25:PRINT"Official reading today," !ys
820 CSIZE 0,0:AT #1,4,25:PRINT"-----"
830 AT #1,17,12:INK 2:PRINT "This bill will be -->"
840 AT #1,17,47:PRINT"Last bill -> £!u + z"
850 PRINT #7," " !s
860 m = r - s
870 AT #1,13,47:PRINT"Units used ->" !m

```

```

880 AT #1,14,47:PRINT"Last time ->" !t
890 AT #1,13,12:INK 2:PRINT"Last reading -->"
900 AT #1,15,47:INK 4:PRINT" -- on"!y#
910 AT #1,9,12:INK 2:PRINT "Standing Charge £"!SCHge/100;"
(units # " !UCHge;"p)"
920 x = INT(UCHge * m)/100
930 PRINT
940 PRINT
950 PRINT
960 PRINT #6," £!x + (SCHge/100)"
970 CLOSE #5
980 DELETE flpl_meter_file
990 OPEN_NEW #4, flpl_meter_file
1000 PRINT #4, r
1010 PRINT #4, m
1020 PRINT #4, x
1030 PRINT #4, y#
1040 PRINT #4, SCHge/100
1050 PRINT #4, s
1060 PRINT #4, t
1070 PRINT #4, u
1080 PRINT #4, v#
1090 PRINT #4, z
1100 CLOSE #4
1110 CLOSE #6
1120 CLOSE #7
1130 CLOSE #8
1140 cue
1150 INK 4
1160 END DEFINE Met
1170 REMark *****
1180 DEFINE PROCEDURE Now
1190 LOCAL r,s,t,u,v#m,x,z
1200 PAPER 0:CLS:INK 4:CSIZE 0,0
1210 CLS #0
1220 OPEN_IN #5, flpl_meter_file
1230 INPUT #5, s
1240 INPUT #5, t
1250 INPUT #5, u
1260 INPUT #5, v#
1270 INPUT #5, z
1280 PRINT
1290 PRINT
1300 PRINT
1310 PRINT
1320 PRINT" As at"!v#:"Bill reading was"!s
1330 INPUT" What is this reading ?"!r
1340 m = r - s
1350 x = INT(UCHge * m)/100
1360 display
1370 CSIZE 0,1:PRINT\ " Cost so far:"
1380 CSIZE 0,0:PRINT " ~~~~~~"
1390 PRINT #8," " !r
1400 PRINT
1410 PRINT
1420 PRINT
1430 AT #1,17,12:INK 2:PRINT "Cost so far is -->"
1440 PRINT #7," " !s
1450 AT #1,6,12:PRINT"This reading -->"
1460 AT #1,6,47:PRINT"Units used ->" !m
1470 AT #1,17,46:PRINT"= £!x!" + St.Ch.£"!SCHge/100
1480 AT #1,13,12:INK 2:PRINT"Last reading -->"
1490 AT #1,14,12:PRINT"(on"!v#;" )"
1500 AT #1,13,47:PRINT"Units used then"

```

```

1510 AT #1,14,58:PRINT"->"t
1520 AT #1,9,12:PRINT "Standing Charge"
1530 AT #1,9,38:PRINT "f";SChe/100;" (units @ ";UChe;"p)"
1540 PRINT
1550 PRINT
1560 PRINT
1570 PRINT #6," C"x + (SChe/100)
1580 CLOSE #5
1590 cue
1600 INK 4
1610 END Define Now
1620 REMark *****
1630 Define PROCEDURE NU
1640 LOCAL f,g,h,i,j,l,m,n$
1650 CLS:CLS #0
1660 DELETE flpl_newtest_file
1670 OPEN_IN #6, FLPI_test_file
1680 OPEN_NEW #7, flpl_newtest_file
1690 REPEAT log
1700 INPUT #6, f
1710 INPUT #6, g
1720 INPUT #6, h
1730 INPUT #6, i
1740 INPUT #6, j$
1750 PRINT #7, f
1760 PRINT #7, g
1770 PRINT #7, h
1780 PRINT #7, i
1790 PRINT #7, j$
1800 IF EOF(#6) THEN EXIT log
1810 END REPEAT log
1820 CLOSE #6
1830 PRINT"\\ " Last reading was "f;g
1840 INPUT" What is present reading ? "l;k
1850 PRINT" Enter time & date (hh:mm Dt Mon Day)"
1860 PRINT" e.g. 09:00 09 Jul Thu"
1870 INPUT" -> "l;n$
1880 l = k ~ g
1890 m = INT(l * UChe)/100
1900 PRINT #7,g
1910 PRINT #7,k
1920 PRINT #7,l
1930 PRINT #7,m
1940 PRINT #7,n$
1950 CLOSE #7
1960 DELETE FLPI_test_file
1970 COPY flpl_newtest_file TO FLPI_test_file
1980 seelog
1990 END Define NU
2000 REMark *****
2010 Define PROCEDURE seelog
2020 LOCAL a,b,c,d,e$
2030 CLS:CLS #0
2031 REMark 123456789 123456789 123456789 123456789 123456789 123456
2040 PRINT" Last this Units Cost of Time
& date"
2050 PRINT " reading reading used units used of
reading"
2060 PRINT "-----
-----"
2070 OPEN_IN #9, FLPI_test_file
2080 OPEN #10, SCR_440x156a50x55
2090 REPEAT viewlog
2100 INPUT #9, a

```

```

2110 INPUT #9, b
2120 INPUT #9, c
2130 INPUT #9, d
2140 INPUT #9, e$
2150 PRINT #10, a,b,c,, "f";d,,e$
2160 IF EOF(#9) THEN EXIT viewlog
2170 END REPEAT viewlog
2180 CLOSE #10
2190 CLOSE #9
2200 cue
2210 END Define seelog
2220 REMark *****
2230 Define PROCEDURE look
2240 LOCAL a,b,c,d$
2250 CLS:CLS #0
2260 OPEN_IN #5, flpl_meter_file
2270 INPUT #5, a
2280 INPUT #5, b
2290 INPUT #5, c
2300 INPUT #5, d$
2310 INPUT #5, za
2320 IF EOF(#5) THEN
2330 e = 0: f = 0: g = 0: h$ = "-- date unknown": ze = 0
2340 GO TO 2430
2350 ELSE
2360 INPUT #5, e
2370 INPUT #5, f
2380 INPUT #5, g
2390 INPUT #5, h$
2400 INPUT #5, ze
2410 CSIZE 0,1:AT #1,2,32:PRINT "last Bill ! "h$(4 TO)!"Bill"
2420 CSIZE 0,0:AT #1,6,31:PRINT "-----"
2430 AT #1,7,14:PRINT "Reading was "a;" ! "e
2440 AT #1,8,44:PRINT "!"
2450 AT #1,9,14:PRINT "Units used were "b;" ! "f
2460 AT #1,10,44:PRINT "!"
2470 AT #1,11,14:PRINT "Cost of Units f;c;" ! f;g
2480 AT #1,12,14:PRINT "Standing Charge f;za;" ! f;ze
2490 AT #1,13,36:PRINT "-----!"
2500 AT #1,14,14:PRINT "Bill total f;c + za;" ! f;g + ze
2510 AT #1,15,44:PRINT "!"
2520 AT #1,16,30:PRINT "as at!"d$;" ! as at!"h$
2530 CLOSE #5
2540 cue
2550 END Define look

```


covered on single screens, such as this one. When no information is available, you get the comments shown in **Figure three**.

New data can be entered in different styles to suit the user's purposes. You can load the help files into word-processing programs, such as Quill, Text87 and Perfection, or text editors such as The Editor or QD5. **Figure four** shows the file

lem, but is rather time-consuming. Help is obtained by clicking the right mouse button, or pressing Enter, with the pointer over the HyperHelp button, dragging the pointer by mouse or keys until the desired function is enclosed by a box, then pressing Enter again. Space scrolls the screen by one line at a time, and Enter by a page. Operations are what experienced Pointer Environment

FLP1_Basic_HELP_ directory for help files, but it can be configured to look at any standard QL device. The ideal would be to have the files installed on a hard disk drive, and copied to ramdisk when the system is booted. You could keep the help disk in FLP2_ if programs and development tools are kept on disk in FLP1_. With ED drives, the greater storage capacity makes it much easier to keep everything on single disks, of course.

Memory needs

The instructions supplied with the program are on five sides of A5 paper, well-printed, and well-written. There is no need for more, since the program is so easy to use.

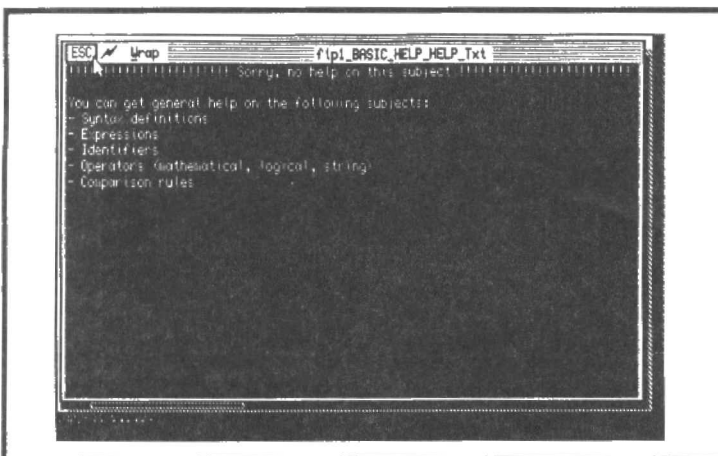
The program disk contains files in both German and English, as usual with Merz Software programs. The Merz "house style" is familiar to me, but some buyers might be confused by the file names on the disk, and the instructions do not explain them. Various program files have _ENGLISH as part of their names; likewise, the file names for the individual help files are English, and these are the ones to copy to your working disk for help displayed in English. The supplied boot routine will load the German version unless you change the filename of the program itself (from HyperHelp to HyperHelp_English) in the boot, or rename the two files themselves.

The program requires roughly 120 KB of memory to run, overall (this figure is not quoted in

the instructions, but is the difference in free memory values before and after the program was loaded). If you do not have the Pointer Environment already on your system boot disk, you need about 35 KB of this for the files PTR_GEN, WMAN, and HOT_REXT. The program's own PE extensions file MENU_REXT requires over 20 KB. The help files, loaded into ramdisk, require about another 140 KB (but you can omit any functions that do not matter to you). You may well have other extensions loaded, for application programs, and/or for hardware devices; for example, The Editor requires 5-10 KB of extensions, the Miracle hard disk drive requires around 50 KB. Clearly, you cannot put all of this on a basic, un-expanded QL

Conclusions

HyperHelp is just the kind of thing relatively inexperienced SuperBasic programmers could use. In fact, many non-programmers would appreciate a quick reference facility for odd toolkit commands. It is not suitable for QLs without memory expansion. A good structure, with reference data which is easy to extend. Average users should find no great difficulties, either in setting-up the program for their particular system configurations, or in adding their own help text. I felt let down in the version reviewed by lack of help on quite a lot of fairly common functions. Merz Software has undertaken to create compatible help files for all its own programs, given time. The price is reasonable.



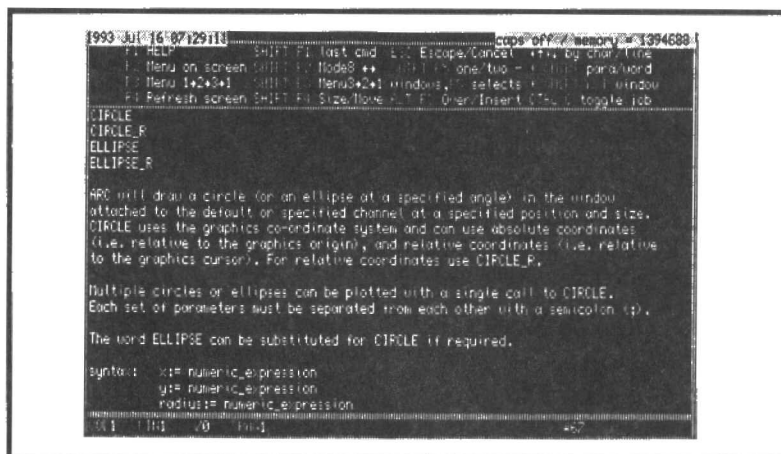
Basic_HELP_CIRCLE loaded into Perfection; compare this with the same data, as displayed on the screen, in **Figure two**. When creating a new help file, add a reference to that file in the index, Basic_HELP_INDEX. Other Merz programs will be supplied with Basic_HELP_INDEX_ADD, covering new functions as well as those already in HyperHelp. The user can load the basic index file into an editor, then append the _ADD index file to the end of it; the whole file can then be sorted and edited, to order everything and remove duplications (if any). Other programs will also have their own help files, to be copied to the HyperHelp file directory. To get help on a specific function, point to it on the screen with the mouse pointer, then press F1.

Keyboard too

A mouse is desirable, but, like most Pointer Environment programs, not essential. Albin Hessler's SERMouse was used during this review, and worked flawlessly. Moving the pointer with the cursor keys is no prob-

lem, but is rather time-consuming. Help is obtained by clicking the right mouse button, or pressing Enter, with the pointer over the HyperHelp button, dragging the pointer by mouse or keys until the desired function is enclosed by a box, then pressing Enter again. Space scrolls the screen by one line at a time, and Enter by a page. Operations are what experienced Pointer Environment

users would expect, and are quite easy to understand for other users. Ctrl-C switches from HyperHelp to another program; when you switch back, the screen display is as you left it, so that you can put new help text on the screen and toggle quickly between it and the program you are writing. When you have had a Gold Card fitted for a year or so, you tend to forget what life is like with a basic QL, but there is no reason to think HyperHelp would be in any way unacceptable - as far as speed is concerned - on the basic machine. However, the program and extensions take a fair amount of space, plus the Help files; you probably won't be able to load the latter into a ramdisk on a basic QL. The review was done with the files left on disk, and the speed of access was quite acceptable. The drive used however was an ED (extra-density) one, which has a much better access time than DD or HD drives. The program defaults to searching the



SuperBasic in Action

Simon Goodwin crafts a Mouse Organ in Amiga SuperBasic

It's rare to find music discussed in QL World. The QL

BEEP function is rather eccentric and other approaches - like MIDI and net sound - have made little impact. But the Amiga Qdos emulator has access to impressive sound facilities, and it is surprisingly easy to use them from SuperBasic.

This month's SuperBasic in Action project generates accurate polyphonic stereo sounds controlled by the mouse. It uses new PTR_ keywords introduced in Amiga Qdos 3.20, and direct access to the Amiga's sampled sound hardware. Unlike QL sound, you can control hardware filtering and edit the waveform and hence the timbre of the music.

Even if you do not have access to an Amiga, the program is interesting as it calculates musical intervals more precisely than the 'even-tempered' keyboard of pianos and other instruments, which are tweaked away from pure ratios to allow them to harmonise acceptably, but not exactly in any one key. Computers can generate more exact chords in all keys than such antique hardware, and it's a pity that most programmers stick to the twelve note chromatic scale when they have access to so much more.

Beethoven is spinning in his grave

I shall discuss music from first principles, with a few references for those with a knowledge of music. I'm grateful to Miles Walsh and my brother Sam for theoretical insights.

Sounds and notes consist of repeated waves. Their frequency or pitch increases as their 'wavelength' decreases, and the waves arrive closer together. Chords are combinations of notes, generally those that beat together harmoniously, generating interference patterns characteristic of the ratio of their frequencies. The waves fit together neatly and the combination of notes sound 'in tune'.

When the higher note has twice the frequency of the lower one, the notes are said to differ in pitch by an octave. So universal is the eight-note scale in western music that the interval between one note and the one at double its frequency is always called an "octave", even when other divisions are used.

Most western music divides the octave into twelve steps of one "semitone" along the piano keyboard in black and white notes. The thirteenth note, "eighth" or "octave" begins the pattern again at a higher pitch. Other pairs of notes use different ratios - for instance a gap of seven semitones on a piano

```
1000 REMark Amiga Qdos STEREO TONE GENERATOR
1010 REMark Uses 8 chip RAM bytes. Version 6
1020 REMark By Simon N Goodwin, 7 July 1993
1030 REMark Uses HEX, ALCHP, RECHP, SYSBASE
1040 :
1050 REMark Pentatonic musical constants
1060 root=2^(1/53)
1070 note2=root^9
1080 note3=root^17
1090 note4=root^31
1100 note5=root^39
1110 DIM note$(39)
1120 here=28800:REMark Or 25920, or 32767
1130 FOR i=39 TO 4 STEP -5
1140   note$(i)=here
1150   note$(i-1)=here/note2
1160   note$(i-2)=here/note3
1170   note$(i-3)=here/note4
1180   note$(i-4)=here/note5
1190   here=here/2
1200 END FOR i
1210 :
1220 REMark Limit of chip RAM
1230 chip_top=2^21
1240 REMark Hardware register labels
1250 pra =HEX("BFE001")
1260 dmacon =HEX("DFF096")
1270 adkcon =HEX("DFF09E")
1280 aud0lc =HEX("DFF0A0")
1290 aud0len=aud0lc+4
1300 aud0per=aud0lc+6
1310 aud0vol=aud0lc+8
1320 aud1lc =aud0lc+16
1330 aud1len=aud0len+16
1340 aud1per=aud0per+16
1350 aud1vol=aud0vol+16
1360 :
1370 REMark Find some chip memory
1380 chip=ALCHP(8)
1390 fast_ram=(chip>=chip_top)
1400 IF fast_ram
1410   RECHP chip
1420   REMark Grab workspace beyond Qdos tables
1430   chip=PEEK_L(SYSBASE+124)
1440 END IF
1450 REMark Prepare simple wave table
1460 RESTORE
1470 REMark DATA -126,-90,-54,-18,18,54,90,126
1480 REMark DATA 127,127,127,127,-127,-127,-127,-127
1490 DATA 0,90,127,90,0,-90,-127,-90
1500 FOR i=chip TO chip+7
1510   READ x : POKE i,x
```

```

1520 END FOR 1
1530 :
1540 POKE_L aud0lc,chip
1550 POKE_L aud1lc,chip
1560 POKE_W aud0len,4
1570 POKE_W aud1len,4
1580 POKE_W aud0per,789
1590 POKE_W aud1per,789
1600 POKE_W aud0vol,64
1610 POKE_W aud1vol,64
1620 POKE_W adkcon,255 :REMark No modulation
1630 :

1640 HELLO : POINT
1650 AUDIO_ON 1+2 : cutoff=0
1660 REPEAT sing
1670   POKE_W aud0per,note%(PTR_X% DIV 6)
1680   POKE_W aud1per,note%(PTR_Y% DIV 6)
1690   k$=INKEY$ : IF k$=" " : EXIT sing
1700   IF k$="#" : cutoff=NOT cutoff : FILTER cutoff
1710 END REPEAT sing
1720 AUDIO_OFF 1+2
1730 IF NOT fast_ram : RECHP chip
1740 CLS #0 : PTR_OFF
1750 STOP
1760 :
1770 DEFINE PROCEDURE POINT
1780 PTR_LIMITS 0,0,239,239
1790 PTR_POS 120,120
1800 PTR_INC 1,1 : PTR_ON
1810 END DEFINE POINT
1820 :
1830 DEFINE PROCEDURE HELLO
1840 REMark Display note grid
1850 OPEN #3,scr_480x240a0x0 : CLS #3
1860 FOR j=6 TO 239 STEP 6
1870   pen=3 + ((j MOD 30)=18)
1880   BLOCK #3,2,240,j*2,0,pen
1890   BLOCK #3,480,1,0,j,pen
1900 END FOR j
1910 REMark Overlay instructions
1920 CSIZE #3,2,1 : OVER #3,-1 : INK #3,7
1930 PRINT #3,," AMIGA QDOS SOUND DEMO"\\
1940 INK #3,4
1950 PRINT #3," Move the mouse to play a stereo sample"
1960 PRINT #3,\\
1970 PRINT #3,"   Top left gives highest pitches"\\
1980 PRINT #3,"   Bottom right for lowest pitches"\\
1990 PRINT #3," Press any key for silent SuperBASIC..."
2000 CLOSE #3
2010 END DEFINE HELLO
2020 :
2030 DEFINE PROCEDURE AUDIO_ON(x)
2040 REMark X =1, 2, 4, 8 for channels 1-4 on
2050 IF x>0 AND x<16
2060   POKE_W dmacon,x-32768
2070 END IF
2080 END DEFINE AUDIO_ON
2090 :
2100 DEFINE PROCEDURE AUDIO_OFF(c)
2110 REMark Turn off audio channels 1, 2, 4, 8 only
2120 IF 1<=c AND c<=15
2130   POKE_W dmacon,c
2140 END IF
2150 END DEFINE AUDIO_OFF
2160 :
2170 DEFINE PROCEDURE FILTER(flag)
2180 IF flag
2190   POKE pra,PEEK(pra) && 253
2200 ELSE
2210   POKE pra,PEEK(pra) || 2
2220 END IF
2230 END DEFINE FILTER
2240 :
2250 DEFINE FUNCTION SYSBASE
2260 RETURN 163840
2270 END DEFINE SYSBASE

```

keyboard comes close to a ratio of 3:2, another pleasing beat known to musos as a 'fifth' (the first and fifth note of the usual scale).

To understand Western music theory you have to remember that a normal scale has twelve notes of which eight are normally used in any one key. The twelve notes start to repeat on the thirteenth, but the eight notes start to repeat on the eighth. Clear as anything, isn't it?

The twelve-note scale on a normal instrument is less accurate for other intervals, like 5:3, 5:4 and 9:8, because it doesn't have enough divisions. This is a shame, as these ratios are common in both Western and Eastern music, and have manifold musical names. But don't be put off. All that matters is the ratio of the wavelengths.

The twelve- (or eight-) note scale gets from one frequency to twice that in twelve equal (or eight unequal) steps, but you only need to stop at five of these steps to generate all the ratios I have mentioned. A five-note scale is known as a pentatonic scale, and pentatonic scales are the basis of much ancient religious music and 20th Century Blues. If you're struggling to play the finale from Pete Townshend's Quadrophonia, try the black notes of the piano keyboard!

(Start from D sharp, and you'll have the pentatonic, aeolian modal or Blues, whichever you like to call it, version of a normal natural minor scale. Start on one of the other black notes, and you'll have something totally different, as the step patterns will be in the wrong place. - triangle-playing Ed.)

This program plays pentatonic music in a two-note chord over an eight-octave range, based on a scale with 53 equal divisions instead of 12. Why 53? Because that works particularly well. You select the notes by moving a pointer across a grid with the Amiga mouse. Red lines separate notes in the scale, and green lines mark each octave.

The highest notes sound when the mouse points at the top left corner. Move it to the bottom right for deep bass. Press # to switch on and off the Amiga's filter, which cuts treble to make the sound less harsh. Press SPACE to stop the sound and return to SuperBasic.

The interesting thing is the variety of chords you get by moving the pointer sideways, up and down. Diagonal positions give the same pitch from each stereo channel, and the notes separate in a pleasing pattern as you move away from that line.

How it works

The program derives note wavelengths from an even scale of 53 notes, rather than the Western twelve-note scale. This would be unfeasible for a piano no-one has long-enough arms - but is no extra trouble for a computer.

Line 1060 sets ROOT to the number which you have to multiply by itself 53 times to get 2, sensibly known as the fifty-third root of two. Replace the value 53 with a temporary variable if compiling with Turbo or Supercharge, or the compiler will 'optimise' the integer expression 1/53 to zero, and all five notes will be the same. Oops.

The next four lines pick the required ratios: 9, 17, 31 and 39 steps along the 53 note scale. If you use a twelve-note scale, use 3, 5, 7 and 10 and replace the 53 with 12. If it sounds better, you're hooked on convention.

The array NOTE% holds ascending wavelength values for 40 notes. You may wonder why I don't just work out the ratios directly. The program evolved to allow easy experimentation with new scales and intervals while staying in touch with Earthly norms. The initial value 28800 was suggested by Sam because its factors are 2, 3 and 5, fitting the scales nicely without rounding errors.

The pointer

The Amiga hardware can move sprite images over the screen under program control, without disturbing the background display. Amiga Qdos lets you turn on a sprite pointer, which subsequently moves around the screen under mouse control as your programs run.

The new functions PTR_X% and PTR_Y% return the position of the pointer within limits set by PTR_LIMITS. Co-ordinates range from 0 to 255 in each direction, like MODE 8 pixels. Line 1780 confines the pointer to an area of 240 units square, matching the grid window #3, opened in line 1850. Six units are allocated to each grid square, for 40 (240/6) notes in all.

PTR_ON and PTR_OFF control the pointer display, while PTR_INC lets you set the step for values of PTR_X% and PTR_Y%. If moving over text in CSIZE 3,1 you might set PTR_INC 8,20 to use character-sized steps.

The hardware

Each Amiga sound channel has four 'ports', which can be programmed with POKES. Lines 1250 to 1350 set standard Commodore port names and addresses. Fourteen bytes control each channel. The first a long word, the address of the wave data that will be played.

The other values are 16 bit words. Next comes AUDOLEN, the length of the wave, in words, 1-65535, followed by AUDOPER which sets the time period between waves, in units of about 280 nanoseconds. The minimum period, or top speed, is about 130 units per byte played - about 28 thousand samples per second - as the data gets transmitted while the display is not busy.

The next word controls the volume, from 0 to 64, so each channel has a 14 bit resolution, with 6 bits of volume control and eight bits of data. One word is unused; then come the registers for the next channel.

The wave data must be stored in 'chip memory', accessible to the custom sound hardware. Most Amigas come with chip memory but you can add 'fast memory' later. If this is present QL tasks and heap can use fast memory, while the system occupies the first part of the chip memory. If ALCHP returns an address below two megabytes, Qdos is running in Chip memory and the space can be used. Otherwise we are in fast memory and need to look elsewhere.

With Qdos running, my 3-megabyte Amiga has 2 megabytes for Qdos code and data, and 800K of spare chip memory; the rest is used for disk track buffers, QL and Amiga screens, system code and variables, and the expansion ROM slot. PEEK_L(SYS-BASE+124) finds the start of this unused space. Amiga Qdos 'roms' like the SER and PAR device drivers sit at the other end. The BOOT program on the Amiga Qdos 3.2 support disk includes code to load toolkits into spare chip memory. If you use this, Mark J Swift recommends CHIP-98688.

SYSBASE, the first address of system variables, is always 163840 on current Amiga Qdos systems. Ignore the last three lines of the listing which set up a Basic equivalent if you use the DIY Toolkit SYSBASE function.

The hardware filter is controlled by an eight bit port, economically known to Commodore as PRA. Unlike the 16 bit ports, you can read as well as write to this, so the program uses PEEK to read the old value before setting or clearing bit 1 to turn the filter on or off.

Two more

Only two more registers need be explained. DMACON controls Direct Memory Access, and the bottom four bits of the word allow the sound channels access to chip memory. The low byte of ADKCON controls special sound effects which combine channels. We POKE

that to 255, to clear it without affecting the other bits, used by the disk system.

How come 255 clears eight bits? The Amiga uses a neat scheme so you don't have to PEEK its word ports before you POKE individual bits. If bit 15 is set when you write to an Amiga word port, all the other set bits in the value you POKE get set in the register. Zero bits are unaffected. If bit 15 is zero, any set bits are cleared in the register. Again, zero bits have no effect. Thus you can whack the bit or bits you want without interfering with the others.

POKE ADKCON with -32767 or -32752 at line 1620 to hear Channel 1 modulating the volume or period of Channel 2. It sounds weird unless Channel 1 is much slower than Channel 2, when you hear conventional analogue synth effects.

The three DATA lines from 1470 to 1490 describe square, sawtooth and rough sine waves. Eight bit sample values range from -128 to +127, and indicate the depth of the wave at successive points. Remove the REMARK before the DATA you wish to use, and RUN the program to hear the new timbre.

If you break in while the sound is playing, type AUDIO_OFF 3 to stop the sound. Channels are numbered 1, 2, 4 and 8 and these values may be used in combination, so AUDIO_OFF 3 turns off channels 1 and 2, the two used by this program. AUDIO_ON 1 turns the first back on again. The direct commands FILTER 0 and FILTER 1 switch the hardware filter off and on, controlling the brightness of the Amiga power light into the bargain.

Further work

Try other wave DATA for a new tone or timbre. Longer patterns allow a more accurate sine pattern, like a flute without the initial breath sound. You can use samples up to 128K long. Lines 1560 and 1570 set the sample length to four words.

Increase the 4 to 12, replace

7 with 23 at the end of line 1500, to read all three DATA lines, and increase the ALCHP on line 1380 to 24 bytes. Now you can play all three DATA patterns in one wave. This will make all the notes deeper unless you tweak the initial wavelength set at line 1120; experiment near the bottom right corner, to hear - and even count - the beats between notes.

Mathematicians and psychologists can investigate the errors in chords for different scales. What is the next best number of divisions to set to get a mathematically perfect pentatonic scale? How does the wave pattern influence perceived pitch?

If you're more practically minded, why not write a SuperBasic procedure to emulate QL BEEP? Start with just the first two parameters, and go on to the others when the pitch and duration match your QL. You could use UNSET (from **DIY Toolkit** Volume Q) to check the number of parameters, and the Timer routines from Volume H for processor-independent timing. The QL waveform is a rough square wave, but you can probably find something more pleasant, if less accurate!

SuperBasic in Action

Easy with Easel -

part 4

Henry Orlowski examines the powerful design features inside Easel.

In April's issue we finished our graph design and examined the different formats in which it could be displayed. This time we will look at the fine tuning we can employ to change almost every aspect of the graph to put a personal stamp on it.

Load up **Easel** and the graphs you saved previously. Make 'boys' the current figures using the View or Olddata command if it is not the current set displayed.

The first bit of fine-tuning we will look at is to make certain parts of the graph stand out. Change to format 0, 1 or 2, the vertical bars. Looking at the bars it will suddenly strike you that it is notable that so many boys liked a cooked meal rather than, say, a quick takeaway, such that it was worthy of highlighting this result. To do this, use the H for Highlight command. Then press the V for value option and select the cell whose value wants highlighting. (You will have noticed that your other option is to select to highlight all negative values, but this is not applicable in this case.) Next you are given the option to select a defined highlighted bar or you can even design your own (more on designing your own bars later on). Try a few different selections and see which is the best for the job.

Now change to format 7 for the pie chart option. You will see what you would have got if you had carried out the same sequence of operations. In this case the segment is highlighted by detaching itself from the chart.

Making Changes

Now let's return to one of the vertical bar formats. You might consider that you don't like the look of the bars you've got. You can change the colour and even give them a border.

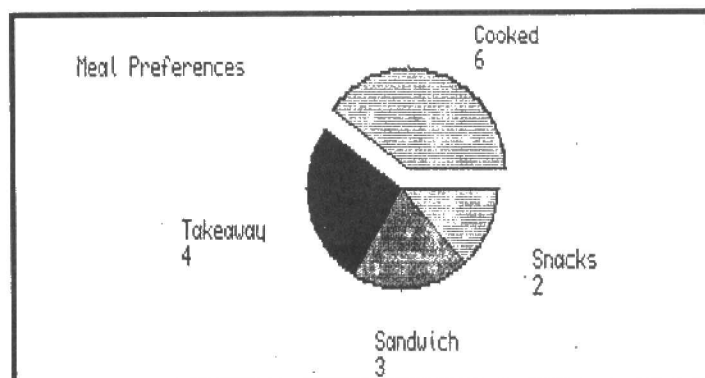
Choose C for Change, then B for Bar. Easel then asks you for the number of the bar you want to change to. Of course, you won't know what corresponds to what, so instead of putting in a number in response, just press Enter at the ? prompt. You will see displayed 18 different numbered bar types on screen for you to choose from. Use the cursor keys to move between bars, then use Enter to make your selection. Try a few out.

If you can't find one that you like, you can actually design your own. To do this select the ? bar option to put yourself into bar design mode. You now have a series of operations you can move between using the up and down cursor keys. Pressing Enter while on an operation allows you to make changes within that operation. The first is the bar colour. Use the cursor keys to move between the four possible options, and note how the diagram on the right changes to reflect the design choices you are making (of course, it helps to have a colour screen).

Then you can select a border colour for your bar. This is the border around the bar itself, and again you have a choice of four. It might be best to examine the next option first, however; this is the border thickness and you should be aware that if your bar has no border then it means

that the border thickness is in fact zero, so changing the border colour will not make any difference. However, entering a border width will make any border colour come into being. The border width is expressed as a

your liking and you want it changed to make some text stand. This does affect the 'title', so enter some text like 'School Market Research Project' and position it where you like. To make changes to this text, use C



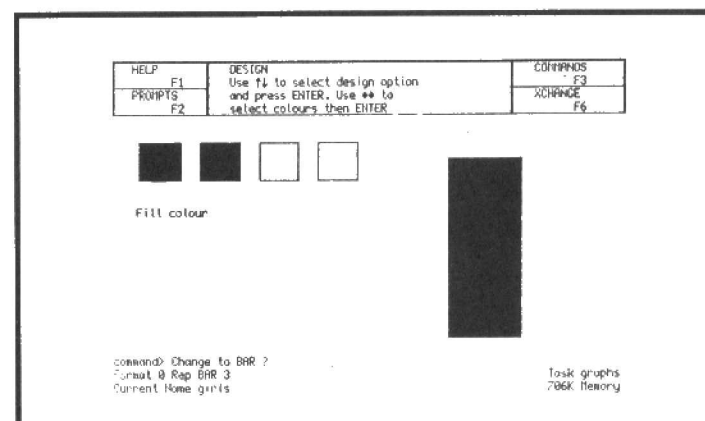
number between 1 and 100. Don't worry what its unit is, just try out a few different values and see what you get. Once you are happy with your design combination press the 'satisfied' option and all will be displayed.

Tweaking text

Moving further on, you might decide that the text style is not to

for Change followed by T for Text. A number of options are then presented to you in the same way as with the bar design operation.

The first is the text colour, followed by the background paper colour. This is not the colour of the whole graph but the area immediately behind the text in question. The next option is to choose a transparent back-



ground paper, in other words, the same as the graph paper background. If you want this option you will notice that it toggles between on and off every time you select it. If on, the background paper colour is ignored. Lastly, you can select a text direction which toggles between vertical and horizontal each time. Why not choose 'vertical' to make it stand out from the title of the graph? When happy with your selection press 'satisfied'. If you don't like what you see, try again.

There are still a number of additional options available to us to change the appearance of our graph further. The major area of the graph is usually the graph paper itself, the area which is currently made up of the background grid on a paper colour you can see on the display.

To change this use C for Change followed by G for Graph. You will see a display of seven graph styles for you to make a preference. Alternatively you can design your own, so why don't you do so. Accept the ? to get into design mode. You navigate between the options and make your selections as before with the bar and text change operations so I won't repeat it. Suffice to say that you now have the ability to design the paper colour and the grid colour by following the indications displayed to you. Experiment a little. It's the best way to grasp it. If you don't like your design when you see it in all its glory then try again until you're happy.

Axis and label

We're not finished yet. You can also change the style of your graph's axis design. To do so you must use the Change command as before, this time with A for Axis. Try it and select one from the 10 styles shown. Or, design your own. The design option lets you come up with your own combination of axis and label colours, axis limits (the low and high values of your axis) or even whether you want the axis lines drawn or not. As before try a few combinations

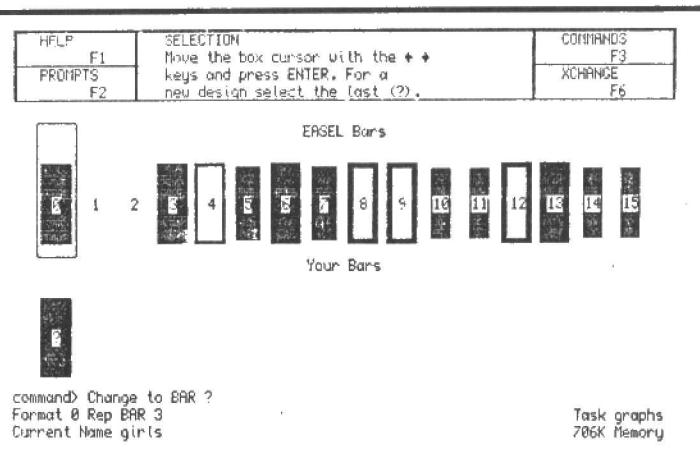
and see how they look.

Let's now look at a further option. Change to format 3 or 4 depending on your version of the program, which is the line graph format. Line graphs are best for displaying trends, but we are experimenting. What we will do now is change the style of the line. A number of possibilities are available. Use the Change command again but this time select L for Line. You can now give a line style number, which you won't be able to do because you are not familiar with the options, or accept the ?. This gives you a display of 16 possibilities to choose from. Or you can design your own. Let's design our own. Move to the ? and Enter.

You are now presented with a number of design options for you to consider. Firstly you have the choice of four line colours for you to select one that looks the best. Then you can choose whether to display a symbol or not. The symbol means a small marker to denote the position of each point on the line. Thirdly, you can choose the colour of the symbol, out of four possibilities, and then you can have your line 'filled' if required. This option gives you a range of mountains rather than just a line and serves to give a sense of power or importance to the values it represents. If, for example, you are displaying two sets of figures, one of which is a set of actual values while the other is the minimum acceptable below which you must not go, then our mountain range filled line could be used to represent the critical base. And finally you can choose the thickness of the line itself within a range of 0 to 100. Again the best thing to do is to experiment and see what combinations you can come up with.

Shades of grey

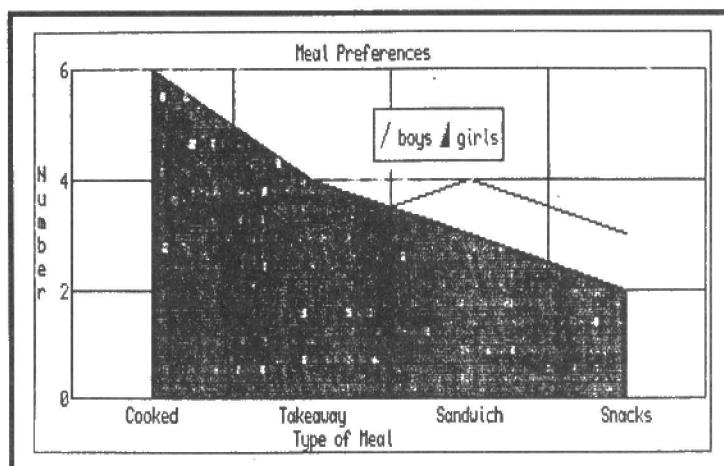
I think you will agree that the sheer scope for customising your graphs is limited only by your imagination. As always you should bear in mind that what you end up with on your monitor will not be accurately repro-



duced when you print it out. A mono printer, as most of us have, will only produce shades of grey to represent different colours, and even a colour printer may have difficulty in differentiating all the different colours you wish to have produced. The important thing is to produce something that has your own personal stamp on it, and that

sets of figures, which you specify separated by commas, yet retains all your settings, including all your design work. So if, for example, you conduct a similar survey at another school, you can just enter the new results into the same basic graph set-up.

Zap is different because it gets you back more or less to the



you should now be familiar with the enormous range of functions that Easel can offer to create new and original offerings for a range of potential uses.

Always remember to S (Save) your masterpieces for future call up so that you can retrieve them at any time. Once you've got your graphs safely tucked up in their disk or microdrive file, and you want to create a new graph of some sort, then you can either K (Kill) or Z (Zap) your existing sets of figures held in memory.

Kill and Zap

If you use K for Kill it deletes all your data from one or more

original basic graph layout you are presented with when you first load up Easel. This includes the original horizontal axis labels, Jan to Dec.

If you want to leave Easel altogether for now then use the Q for Quit option.

Finally, I would like to thank you for following this series of articles. I trust you have found it enjoyable and informative. My aim was to make you aware of Easel's potential for everyday applications in addition to the more obvious business ones, and if I've made you stop to think about it a little then I have been on the right road. Stay with it a little longer - it's an opportunity not to be missed.

Data- DESIGN V3

Wolfgang Lernerz has been using and commenting on the new DataDesign.

The database market for the QL isn't really saturated. Apart from the ubiquitous Archive, the only serious contender was Flashback, which had the advantage of being faster, but could not be programmed, and its report generator left much to be desired. Now there is DataDesign, a database management system for the QL, programmable in 'C', Basic and Assembler.

Perhaps I should set out right from the start that I have been slightly "involved" in this program as a beta tester. Having said that, I have no connection to Progs, the authors, other than being a (difficult to satisfy...) user of their program. It is probably due to the fact that I didn't stop bombarding them with suggestions that they asked me to beta-test version 3 of their programs. On the other hand, I can witness their willingness to listen to users, and make reasonable amendments/additions. Progs' after-sales service leaves nothing to desire! I do admit, however, that I am really taken by this software, which I find very powerful.

More friendly

DataDesign 3 is (you might have guessed) a successor to DataDesign 2, a moderately successful database management system (DBMS) on the QL. To exist next to Archive, which comes free with every QL, a DBMS must be faster, more powerful and more friendly than

its eternal rival. I find that DataDesign is. The interesting thing about DataDesign is that it comes in three parts: first, there is the database "engine", the part that actually handles all the data. This is the most important part of the system. Second, there is a badly-named 'Main program', which is actually a front end to the engine, allowing easy access to the engine and the data. Most users who are not programmers will use this front end to access the database. Finally, there is the "Application Programmers Interface" (API). This is the description of how you can access the database from within your own programs, enabling you to program the engine directly.

This means that DataDesign will be suitable at the same time for those who want a simple, no-hassle, cardfile-type database (a la Flashback), and for those who want to write fully-fledged database applications. Also, as both are sold separately, you can buy only that part which you really need. I shall concentrate first of all on the engine itself, then on the front end and finally on the API.

The engine

The engine is the very heart of the DBMS. It is through this that you access the data, add to it, delete from it, filter or sort it, etc. It is also the engine which determines whether a database is fast, how many records can be used, how the database is

organised and so on. The distinction between the engine and the "front ends" exists in nearly every database (even in Archive). DataDesign is special in that the distinction between the engine and the various front ends is made quite clear. The DataDesign engine has two fundamentally different ways of handling files: files can be disk-based, or memory-based. The disk-based approach is that used by Archive: all the data, except for a few records, remains on the disk and is loaded only for displaying (or selecting, sorting, etc.). The other way of handling data, which can also be implemented in DataDesign, is to load all records into memory right away and keep it there until one quits the program.

Both methods have their advantages and disadvantages: leaving the records on the disk will slow down access to the data, as it will have to read from the disk each time it is to be accessed. Having the data in memory makes access to it is so much faster. On the other hand, in theory a disk-based database is safer: if the computer crashes, only one record should be corrupted, that currently being updated and which hasn't been saved yet. In a memory-based database, unless the data is explicitly saved after each modification, it will all disappear if the computer crashes (as has been known to happen to the QL!). In this, Archive is not typical, because it has both disadvantages: it is disk-based, and slow, but if the computer crashes with the file

still open, you still lose all your data. They got it wrong there!

This does not happen with a disk-based file in DataDesign, where at the utmost you lose only one record (the one in memory, or the one being written out to the disk). DataDesign is noticeably faster than Archive even in disk-based mode.

Memory size

The DataDesign engine also accepts that a file can be memory-based, increasing yet again the speed of access. Of course, when a file is memory-based, all data is held in memory, so that, with a large database (over 700 KB, as are some of mine), one might need at least a Gold Card.

Like every other database, DataDesign is organised in records, each record containing a given number of fields. According to the documentation, DataDesign has no limit on the number of records it can handle, except for memory: in a memory-based database, the number of records will depend directly on the amount of data contained in each record. For a disk-based file, each record takes 18 bytes of memory at least (this can be more, if different indexes are used; see below). Here the power of DataDesign becomes apparent, as it is possible to switch between disk-based and memory-based operations at any time! If a memory-based file becomes too large, one can

make it disk-based, and continue working on it.

Each record can be subdivided into fields, up to 256 fields per record. That should be enough for most of us. If not, it is possible to sub-divide each field into lines and, of course, access each line independently. Moreover, the lines of the fields have no imposed maximum length, they are as large as the data put into them. This, combined with the fact that DataDesign accepts "raw" fields (which have no determined type and can be used, for example, for images), means that DataDesign is a free-format database. At the same time, the DataDesign engine allows you, should you so wish, to give a determined type to each field - the allowed types are: raw (anything), string, integer (2 bytes), long word (4 bytes) and IEEE double (8 bytes).

Fresh fields!

Concerning the fields, a significant difference with Archive can be seen: it is possible, at any time, to add (or delete) a field! This is not possible with Archive where one has to create a new database containing all of the old fields as well as the new one, and then copy all the data across.

The DataDesign engine can access an unlimited number of files at the same time, so there can be an unlimited number of files 'open' and, of course, data can be passed from one file to another. In that sense, DataDesign is a true relational database. At the same time, an unlimited number of jobs can call upon the DataDesign

engine simultaneously - two (or more) jobs can even work upon the same database at the same time! As this can even take place over a network (!), DataDesign is a true multi-user database. Of course, a sophisticated 'record-locking' system has been implemented for this, to avoid having one job/user accessing a record which is currently being modified by another job/user.

DataDesign is quite fast when ordering a file, or filtering it (select in Archive). Moreover, it is the only QL DBMS I know of that implements indexes. An index is nothing more than a way of describing how a database should be ordered/selected. When one orders a file in Archive in a certain way, an index of where every record should be in that order is built up. The same is true in DataDesign. However, if one

figure 1: The main program - field names on the left, contents on the right

F1 Help	F2 File	F3 Command	F7 Display	F8 Status
INTfield	1			
LONGfield	24000001			
DATEfield	1017502495			
FLOATfield	1.003259			
field 5	rec1- field5			
field 6	rec1- field6			
field 7	rec1- field7			
field 8	rec1- field8			
field 9	rec1- field9			
field 10	rec1- field10			
field 11	rec1- field11			
field 12	rec1- field12			
field 13	rec1- field13			
field 14	rec1- field14			
field 15	rec1- field15			
field 16	rec1- field16			
field 17	rec1- field17			
field 18	rec1- field18			

QUIT	DO	Filter file..
+	+	+
Level 1	+	Place:
+	+	+
And	As string	
Mark	Case dependant	
Not	Cleared	
+	+	+
Only previous levels used	+	+
+	+	+
<	=	>
Value :		

figure 2: The filter menu - upto 10 filter levels can be set

QUIT	Command
Begin/first	Duplicate record
End/last	Undo/Truncate
Search..	Delete record
Replace..	Clear all mark
Filter file..	New record
Sort file..	Set MEMO field..
New field..	Clear field..
Erase field..	View file..
Stuff record z..	Quit program

figure 3: The command menu. Other commands are available in the Files menu

needs a new order in Archive, the old one is destroyed, and to get it back one must again re-order the file. This is not so in DataDesign, where one can keep each index (and even save it onto disk) and access the database through these indexes at any time. It thus becomes possible to have several indexes at once in the machine, allowing accesses to the data according to varying sort and filter criteria.

As can be seen from the above, the DataDesign engine is very powerful. It is fast, too.

The front end

To take advantage of the DataDesign engine, one must have a front end, either a

program built by yourself, or the "main program" supplied by Progs. This front end to the engine presents itself as a pointer-driven program (see figure one). Anybody who uses software under the pointer environment will be immediately familiar with the software and its menu system.

The "main program", as it is also called, is designed to be as simple as possible. It hides most of the complexities of the DataDesign engine, at the cost of not being able to use all of its features. It turns the Database engine into a simple flatfile DBMS - one file at a time only, which generally is sufficient for most users anyway. This again is an advantage over Archive

There is not much else to be said about the main program, which admittedly uses the engine at less than half its power. On the other hand, it will satisfy most users (see figure three, some of the commands available) - and those who need more are probably those who would program the engine directly anyway.

It is possible to use the engine - and access the data - directly from your own programs. This is done either via new

For the technically minded, the engine itself is implemented as an extension "thing". When

For those programming in C, a library of function calls is supplied, opening all possibilities of the engine. It is thus very easy to incorporate all calls to the engine into one's own programs.

Of course, programming the engine needn't be very extensive. It is a relatively simple affair to write a (small!) program for, say, a report generator. An advantage of DataDesign is that those who do not feel the need

In summary, I definitely give DataDesign a 'thumbs up'.

Price: DataDesign 3 (engine and main program): 3000 Belgian Francs; DataDesign 3 API: 1000 Belgian Francs.

quality product, British-made and comes with full binding instructions. It is manufactured in a rich, deep blue with genuine gold blocked lettering. Enhance your Sinclair QL World Magazine collection now for only £6.95 (inc.P&P!) *Send for one today! The QL World binders also make an ideal gift for other Sinclair users too!*

Tel No.

Part 4

In part 4, Alan Bridewell deals with repeating functions.

In parts 2 and 3 of this series, we produced loops of code which were designed to be repeated a fixed number of times. So we used a 'loop counter' to count how many more times it had to repeat before it had finished.

But there are many situations where you cannot know, in advance of running the program, just how many times you want to repeat the loop. What you want is for the loop to keep repeating until a certain situation occurs which tells you it has done enough. For example, if you are loading a document into a word processor, you want the program to keep loading until it reaches the end of the document, without it being told first exactly how long the document actually is.

Status register

The microprocessor in the QL has a special register for this purpose called the 'status register'. We have already used this register in the DBRA instruction, but it has been used in such an automatic and controlled way we did not actually need to mention it. But to find out if a particular condition has occurred, we need to find out what is actually in the register, so we need to know something about it.

The most important point about this register, which is completely different from all the others, is that the individual digits it stores are NOT used as one number, but are used to store quite separate bits of information. These digits (normally called 'flags') become 'set' (equal

to 1) or 'clear' (equal to 0)
depending on the result of cer-
tain things the program does.

There are five flags in the register which we need to concern ourselves with, and understand what it means when they become 'set' or 'clear'.

The 'Z' or 'zero' flag is set if an instruction results in a zero being produced. This will happen, for example, when a number is subtracted from an equal number.

The 'N' or 'negative' flag is set if an instruction results in a negative number being produced, for example, subtracting one number from a smaller number.

The 'C' or 'carry' flag is set if an instruction results in a number too big for the register, so a

'carry' to the next digit is needed. This can result from addition or multiplication.

The 'V' or 'overflow' flag is set if the instruction cannot be carried out with the numbers given to it, for example, if we try to divide by zero.

The 'X' or 'extended' flag is set or cleared when certain types of add, subtract, shift or rotate instructions move data in or out of it. This is definitely not beginners stuff, so we shall leave it for the time being.

A full explanation of what does or does not alter these flags really requires a programmer's reference manual, but in general terms it is fair to say that if an instruction MIGHT cause a flag to set, then it WILL set if the con-

dition occurs and it WILL clear if the condition does not occur. If the instruction could not produce the condition, it will leave the flag unchanged.

Branching

There are different ways to use the information stored in the status register, but the most common, and the one we shall look at, is by using 'conditional branching'. There are actually 14 of these conditional branching instructions, each of which will cause the program to branch to a specified address if a particular combination of conditions occurs. Some of these combinations are quite complicated involving 'AND' and 'OR'. We shall deal with two simple,

Listing 1

```

; *****
;      DRAWING LINES USING INTEGER ADDITION AND MULTIPLICATION
; *****
;
; D1 HOLDS THE WORD TO PUT INTO SCREEN ADDRESSES
; D2 HOLDS THE SCREEN WORD ROW NUMBER (UP TO $FF)
; D3 HOLDS THE SCREEN WORD ROW INCREMENT VALUE (UP TO $FF)
; D4 HOLDS THE SCREEN WORD COLUMN NUMBER (EVEN NO, UP TO $3E)
; D5 HOLDS THE SCREEN WORD COLUMN INCREMENT NUMBER (EVEN NO, UP TO $3E)
; D6 HOLDS THE CURRENT SCREEN ADDRESS RELATIVE TO $20000
; A0 HOLDS THE CURRENT ABSOLUTE SCREEN ADDRESS
;
; D1, D2, D3, D4, D5 ARE TO BE SET BY THE 'CALL' COMMAND FROM SUPERBASIC
;
.LOOP      MOVEA.L    #$20000,A0      ; RESET SCREEN ADDRESS
           MOVEQ     #$0,D6          ; RESET RELATIVE SCREEN ADDRESS
           ADD.W      D3,D2           ; INCREMENT ROW
           CMPI.W     #$FF,D2        ; ARE WE PASSED LAST ROW?
           BPL        EXIT           ; YES, THEN EXIT LOOP
           CMPI.W     #$0,D2         ; ARE WE PASSED FIRST ROW?
           BMI        EXIT           ; YES, THEN EXIT LOOP
           ADD.W      D5,D4          ; INCREMENT COLUMN
           CMPI.W     #$3F,D4       ; ARE WE PASSED LAST COLUMN?
           BPL        EXIT           ; YES, THEN EXIT LOOP
           CMPI.W     #$0,D4        ; ARE WE PASSED FIRST COLUMN?
           BMI        EXIT           ; YES, THEN EXIT LOOP
           ADD.W      D2,D6          ; ROW NO, IN D6
           MULU       #$80,D6       ; ROW NO. X BYTES PER ROW
           ADD.W      D4,D6          ; ADD COLUMN
           ADD.W      D4,D6          ; AND AGAIN FOR WORD LENGTH DATA
           ADDA.W     D6,A0          ; ADD TO SCREEN ADDRESS
           MOVE.W     D1,(A0)       ; LOAD WORD INTO SCREEN ADDRESS
.EXIT      BRA.S      LOOP
           MOVEQ     #$0,D0          ; NO ERROR
           RTS                ; RETURN TO SUPERBASIC

```

obvious, and useful ones.

'BMI' stands for 'branch if minus', and branches if the N flag is set, indicating that a minus number has been produced.

'BPL' stands for 'branch if plus', and branches if the N flag is clear, indicating that a minus number has not been produced.

Although we shall not be using them here, I will mention two others, which are also very useful.

'BEQ' stands for 'branch if equal', and branches if the Z flag is set, indicating that a zero has been produced.

'BNE' stands for 'branch if not equal', and branches if the Z flag is clear, indicating that a zero has not been produced.

subtract the immediate data from the data we are comparing it with, and alters the flags according to the result. For example, consider the instruction

CMPI.W #\$FF,D2

Remember the hash (#) tells us we have an actual number, not an address, and the S means it's a hexadecimal number. So the instruction will subtract #\$FF from the contents of data register D2 to see what happens. (The result of this subtraction is not stored, and D2 remains unaltered. The only affect is on the flags.)

If D2 contains a number less than #\$FF, then we get a minus number. This will set the N flag

tions together is a very powerful way of controlling what a program does, and is well worth a lot of effort to master.

Before we get round to actually using these instructions, I would like to introduce the arithmetical instructions.

'ADD' means simply that. It has a few variations like most instructions, but the most important point is that we can add minus numbers as well, resulting in a smaller number. 'SUB' for 'subtract' is just as easy.

There are two forms of the multiply and divide instructions, called 'signed' and 'unsigned'. Microprocessors have a special way of dealing with whether a number is plus or minus, which we shall not deal with here.

When this special way is being used the numbers are said to

be 'signed'. Our last two programs have used signed numbers, but because our assembler can deal with them automatically, we have not had to mention the fact. But signed numbers need special instructions, 'MULS' and 'DIVS', standing for signed multiplication and division.

If we ignore

signs, and treat all numbers as positive, we can use 'MULU' and 'DIVU', standing for unsigned multiplication and division. MULU will multiply two word sized numbers to give a long word sized number.

Blob time

To illustrate this new material, we shall put together a routine to print rows of blobs on the screen. They can start anywhere on the screen, can go in any direction, and will stop at the edge of the screen. But before we can do this, we need to know a little bit more about how the screen ram is converted to dots on the screen.

The information for a row of pixels is held in 128 consecu-

tive bytes. Each word (two bytes) contains the information for either four pixels in 8 colour mode, or eight pixels in four colour mode. Exactly how the different pixels are dealt with within the word we shall leave for the present. We shall simply regard each word as dealing with a long, flat, multicoloured blob, which is actually a short row of pixels. There are 64 such blobs in each row, with 256 rows in a screen.

Clearly, each time we increase the screen address by two, we move right to the next blob. To move to the next blob down, we have to increase the screen address by 128 (which is #\$80 in hexadecimal). This gives us a formula for calculating the address of any particular blob on the screen, as follows.

1. Multiply the row number (from the top) by #\$80. This gets us to the correct row.

2. Add twice the column number (from the left). This gets us to the correct column. It has to be twice the column number because each blob is a word (or two bytes) long.

3. Add this to the address of the start of the screen RAM (\$20000).

For example, suppose we want the address of the blob which is two rows from the top and three columns from the left. The formula gives

$(2 \times \$80) + 6 + \$20000 = \$20106$

Don't worry if you can't do the hexadecimal calculation yourself - the microprocessor will do it for you as long as you give it the correct numbers!

Using CALL

One last point before we look at the actual routine. Clearly it would be very clumsy if we had to rewrite the program every time we wanted to draw a different line. The best thing is to put the data for the line into some registers, then access the data to draw the line we want. The SuperBasic command CALL enables us to do this.

Up to now we have used the

Listing 2

```
100 z = RESPR(512)
110 LBYTES f1p2_LISTING1_code,z
120 OPEN#3,con_512x256a0x0
130 REPEAT loop
140 CLS#3
150 INPUT#3, "STARTING ROW? (0 to 255)      ",row%
160 INPUT#3, "INCREMENT ROW BY? (0 to 255)   ",r_inc%
170 INPUT#3, "STARTING COLUMN? (0 to 63)     ",col%
180 INPUT#3, "INCREMENT COLUMN BY? (0 to 63) ",c_inc%
190 INPUT#3, "WORD TO PUT INTO SCREEN ADDRESSES", word%
200 CLS#3
210 CALL z, word%, row%, r_inc%, col%, c_inc%,0,0
220 PAUSE -1
230 END REPEAT loop
```

You might be wondering what being equal has to do with the Z flag. The answer is quite simply in the most common way of using the status register.

Although most instructions will alter flags in the register, and we can use them at almost any time, what we most frequently do is use instructions whose whole purpose is to alter the flags to give information. These are the 'compare' instructions, which do just that. We shall look at one of these in detail, to see exactly what it does.

'CMPI' stands for 'compare with immediate data'. Immediate data means that the actual number used for comparison is in the instruction, not in some register or memory location from where it has to be fetched first. What the instruction does is to

and clear the others.

If D2 contains a number greater than #\$FF, then we get a plus number. This will clear all the flags.

If D2 contains exactly #\$FF, then we get zero. This will set the Z flag and clear the others.

We can now see what being equal has to do with the Z flag. If we compare two numbers by subtraction, we get zero if they are equal (so the Z flag is set), but we get either a plus or a minus number if they are not equal (so the Z flag is cleared).

Compare and branch

This use of 'compare' and 'conditional branching' instruc-

CALL command with just one parameter, the address we want to start the machine code from. But CALL will take up to 13 extra parameters, and these will be loaded consecutively into the registers D1 to D7 then A0 to A5, before starting from the address given. This is just what we need to load our data into registers before drawing the line.

Listing one shows our routine. It would probably be advisable to follow the listing while reading through the next bit, so that it is clear what part of the listing each point refers to.

As you can see, it expects certain registers to be loaded with data before it starts, and this will be done from SuperBasic with the CALL command as indicated. Address register A0 will be used to hold the ram address of the blob to be drawn, so the loop begins by resetting this to #20000, which is the start of the screen ram. Data register D6 will be used to calculate the relative screen address of the blob, so we clear it to zero at the start of each loop.

D2 holds the screen row of the blob and D3 holds the row increment value (how many rows between each blob). So we increase (or decrease) the number in D2 by the number in D3. Before we go any further, we need to know if this new row number is still on the screen. By comparing D2 with #255 (255 in decimal) we can see if we are off the bottom of the screen. If the N flag is not set, then we are off the bottom, and the BPL instruction branches out of the loop. Similarly, if we compare D2 with #0, the N flag set (meaning a minus result) would cause the BMI instruction to branch out of the loop.

In exactly the same way, we can increment the column by increasing (or decreasing) the number in D4 by the number in D5. By comparing D4 with #31, a BPL will branch out of the loop if we are off the right of the screen, and by comparing D4 with #0, a BMI will branch out of the loop if we are off the left of the screen.

Listing 3

```

100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
130 CLS: RESTORE :READ space:start=RESPR(space)
140 PRINT "Loading Hex...":HEX_LOAD start
150 INPUT "Save to file...":f$
160 SBYTES f$,start,byte:STOP
170 :
180 DEFine FuNction DECIMAL(x)
190 RETurn CODE(h$(x))-48-7*(h$(x)>"9")
200 END DEFine DECIMAL
210 :
220 DEFine PROCedure HEX_LOAD(start)
230 byte=0:checksum=0
240 REPEAT load_hex_digits
250   READ h$
260   IF h$="*":EXIT load_hex_digits
270   IF LEN(h$) MOD 2
280     PRINT "Odd number of hex digits in: ";h$
290     STOP
300   END IF
310   FOR b=1 TO LEN(h$) STEP 2
320     hb=DECIMAL(b):lb=DECIMAL(b+1)
330     IF hb<0 OR hb>15 OR lb<0 OR lb>15
340       PRINT "Illegal hex digit in: ";h$:STOP
350     END IF
360     POKE start+byte,16*hb+lb
370     checksum=checksum+16*hb+lb
380     byte=byte+1
390   END FOR b
400 END REPEAT load_hex_digits
410 READ check
420 IF check<>checksum
430   PRINT "Checksum incorrect. Recheck data. ":STOP
440 END IF
450 PRINT "Checksum correct. Data entered at: ";start
460 END DEFine HEX_LOAD
470 :
480 REMark Space requirements for the machine code
490 DATA 64
500 :
510 DATA "207C00020000": REMark .LOOP      MOVEA.L    ##20000,A0
520 DATA "7C00": REMark                   MOVEQ      ##00,D6
530 DATA "D443": REMark                   ADD.W       D3,D2
540 DATA "0C4200FF": REMark                CMPI.W      #$FF,D2
550 DATA "6A00002C": REMark                BPL         EXIT
560 DATA "0C420000": REMark                CMPI.W      #$00,D2
570 DATA "6B000024": REMark                BMI         EXIT
580 DATA "D845": REMark                    ADD.W       D5,D4
590 DATA "0C44003F": REMark                CMPI.W      #$3F,D4
600 DATA "6A00001A": REMark                BPL         EXIT
610 DATA "0C440000": REMark                CMPI.W      #$00,D4
620 DATA "6B000012": REMark                BMI         EXIT
630 DATA "DC42": REMark                    ADD.W       D2,D6
640 DATA "CCFC0080": REMark                MULU        #$80,D6
650 DATA "DC44": REMark                    ADD.W       D4,D6
660 DATA "DC44": REMark                    ADD.W       D4,D6
670 DATA "D0C6": REMark                    ADDA.W      D6,A0
680 DATA "30B1": REMark                    MOVE.W      D1,(A0)
690 DATA "60C4": REMark                    BRA.S        LOOP
700 DATA "7000": REMark                    .EXIT
710 DATA "4E75": REMark                    MOVEQ      ##00,D0
720 DATA ":", "4658"                      RTS

```

Blob control

By using these compare and branch conditional instructions, we have arranged for the routine to leave the loop as soon as our blob goes off any edge of the screen.

If our blob is still on the screen, we now have to calculate its address, using the formula, as follows:

1. Add the new row number in D2 to D6.
2. Multiply D6 by #80 to get to the right row.
3. Add the new column number in D4 to D6, but do it twice, because each blob is a word (2 bytes) long.

D6 will now hold the address of the blob relative to the screen start address (#S20000). To get the absolute screen address of the blob into A0, we add D6 to the #S20000 already in A0. We can now print the blob on the screen by moving the blob data, held in D1 into the address held in A0. We end the loop with a

BRA instruction to start the loop over again.

When we finally leave the loop because the blob has wandered off the screen, we must tidy up by putting zero in D0 to make sure we get no error message, and then return to SuperBasic with RTS.

Listing two shows a SuperBasic program to load and run the routine. It is designed as an endlessly repeating loop to input the parameters from the keyboard to see what they produce on the screen. Once you have got it working, and feel you understand what is going on, it would be far more interesting to rewrite it to draw lines automatically, either at random, or in a fixed pattern. If you do not CLS between each CALL, you could draw patterns, boxes, etc. By using CLS, and arranging for the parameters to change slowly and smoothly, you could make the line appear to move smoothly round the screen. Adding a random colour change to this, could produce some interesting effects. In

mode 8, you can also get flashing as well. All these involve quite elementary SuperBasic programming techniques, and which I won't elaborate on here.

Suggestions

When you feel confident enough to start playing around with the assembler listing, here are some suggestions.

This loop could be embedded in a larger loop which altered one or more of the parameters for plotting the line. In this way you could have several lines plotted with one CALL of the code. This outer loop could be another conditional loop, only stopping when parameters reach certain values, or it could repeat a fixed number of times. In fact a whole series of embedded loops could be put together, each one involving the change of one parameter. This is not difficult, provided a couple of precautions are taken. First, each loop must have a different label for its own start and its own exit address. Secondly, you

must take care about incompatible branching conditions, which may result in the loops never starting, or never stopping!

Instead of letting the lines stop at the edge of the screen, you could stop them before they get there. By using the spare registers to hold the row and column numbers to stop at, you could put those values into the compare instructions. In this case it would be a CMP, not a CMPI instruction, because you would be comparing with a value in a register, and not with immediate data. For example, suppose D0 held the row you wished to stop at. You would use the line

CMPW D0,D2

followed by either a BPL or a BMI, depending on whether this was the furthest right or left you wish to go. As before I have included the code in Marcus and Simon's Hex Loader for those who do not have an assembler. This is Listing three.

Happy coding!

TF SERVICES

MINERVA

The ULTIMATE operating system upgrade

MKII MINERVA with battery for 256 bytes ram, CRASHPROOF clock & Philips PC bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

Other features common to MKI/MKII: DEBUGGED operating system/ autoboot on reset/ Multiple Basic/ faster scheduler-graphics (10% of lighting)-string handling/ WHEN ERROR/2nd screen/ TRAC/ foreign keyboard drivers/ "warm" fast reset/ Auto reboot after power failure. V1.97 now with built in Multibasic and split OUTPUT baud rates with Hermes.

1st upgrade: free. Otherwise £3 (+ £5 for manual if req'd) (send sse, rom & NEW disk/3 mdvs)

MKI to MKII upgrade - £30

MKI.....£40

MKII.....£65

BOTH VERSIONS GOLD CARD COMPATIBLE

HERMES

A replacement QL co-processor for the QLS awful IPC 8049

- Do you get keyboard bounce?
- Do you find fast serial input unreliable?
- Do you want to connect a modem at 19200bps

If you can say YES to any of these, then you need HERMES

- 19200bps RELIABLE serial input - NO QCONNECT.
- Independent input baud rates - use serial mouse & print
- Stops keyboard bounce (unwanted repeat chrs)
- Improves "fuzzy" and "random" sound
- Provides extra input/output lines
- Key click

Fitting is simple. Remove the QL top (8 screws) & replace the chip marked 8049 or 8749 next to mdv 1.

£25 including manual/software

QL SPARES

Faulty QL board (no plug-in chips).....£9

Keyboard membrane.....£9 Circuit diagrams.....£2
68008 cpu.....£8 1377 PAL.....£3
IM ROM.....£10 Power supply.....£12
8302 ULA.....£10 8301 ULA.....£10
8049 IPC.....£8 MDV ULA.....£12

Other components/sockets etc) please phone

QL REPAIRS

Fixed price for unmodified QLS, excluding microdrives. QLS tested with Thorn-EMI rig and ROM software

£27 including 6 month guarantee

All prices include post & packing (UK only). Payment by Mastercard/Visa/Eurocard/cheque/postal order/PO Giro transfer (58 267 3909). MAIL ORDER ONLY - no callers without ringing first. Ring for overseas prices.



12 Bouverie Place, LONDON W2 1RB

Tel: 071-724 9053

Fax: 071-706 2379



Telephone: 0753-888866

Fax: 0753-887149

(EEC)

W.N. Richardson & Co.

18-21, Milsbourne House,
Chalfont St Giles, Bucks

SINCLAIR QL PRICE LIST MARCH '93

All Prices Include 17.5% VAT

QL Computers

NEW MONITOR £50 CHEAPER

JS COMPLETE

QL Computer, PSU, T.V. Lead, PSION V2.35 Software (Word Processor, Spreadsheet, Database & Business Graphics), Handbook for QL Programs, & Superbasic. JS £120

Free one years membership to QUANTA, the Independent QL User Group with over 400 free library programs, Newsletters & HELP. 6 MONTHS WARRANTY. WITH IM ROM £100.00

PART EXCHANGE £30 OFF ABOVE. Send in old QL (Unit only, any condition) - JS ROM £90 IM £70.00

BACKUP QLS

QL & PSU only. JS £80 IM £65 (Part exchange allowance £15 if required)

Accessories

NOTE: EXTERNAL (SER2) 3 BUTTON MOUSE AND SOFTWARE WITH EXTRA FUNCTIONS. NOW HERMES COMPATIBLE

PC KEYBOARD INTERFACE, INTERNAL FITTING, FOR FITTING 102 KEY KEYBOARD £75.00
PC KEYBOARD, UK version, 102 key (AT) £30.00
PC KEYBOARD INTERFACE AND KEYBOARD (INCLUDES FREE JOYSTICK OR PSU) £95.00
CABLE AND LEAD FOR EXTERNAL FITTING of Keyboard Interface £14.00
JOYSTICK with QL lead (no interface required) £10.00
* QL MOUSE, 3 Button, Software controlled, externally mounted, simply fits in SER2 £45 CORDLESS MOUSE £55.00

Disk Drives

THE UNIVERSAL DRIVES ARE ALSO SUITABLE FOR ACORN BBC, ATARI ST, AMIGA, SPECTRUM, IBM, AMSTRAD AND OTHER PC COMPATIBLES

Universal 3.5" 1Mb disk drive, cased with PSU and free QL lead £70.00 2 Units for dual drive £120.00
Universal 3.5" 2Mb disk drive, as above but 2Mb capacity £90.00 2 Units for dual drive £170.00
3.5" 1Mb Uncased disk drive £25.00
3.5" 2Mb Uncased disk drive £29.00
Lead for uncased disk drives, or other universal micro £10.00 POWER SUPPLY FOR UNCASED DRIVES £6.00
METAL CASES FOR UNCASED DRIVES £3.00

Parallel Printers

NEW RANGE

SAMSUNG SP093 80 Col, 300 cps, 50 NLO, 3K Print Buffer, Canonics, Tractor Feed/Sheet Feed, Paper Park OLIVETTI COLOUR PRINTER, 24 pin, 200 cps, 50 NLO, 40K Buffer, Epson LQ2550/IBM Compatible CENTRONICS PARALLEL Printer interfaces for above printers £149.00
QL LEAD FOR SERIAL PRINTERS £10 £27.00
CANNON BJ10 EX £225

Monitors

PHILIPS 14" HIGH RES COLOUR EGA 31 DOT PITCH, FULL 85 COL WITH AMBER OR GREEN TEXT FEATURE, IDEAL FOR WORD PROCESSING, RECONDITIONED, 90 DAY WARRANTY. £149
OK FOR PC AND QL SELF SENSING AND MANY EXT'L CONTROLS

Microdrive Cartridges and Spares

"MONO OPTION" WITH 9" GREEN SCREEN £39

4 New Cartridges in a wallet £10.00
Plastic Storage filing box including 20 new cartridges £45.00
QL Psion Software V2.35 Includes Quill, Abacus, Archive, and Easel IN WALLET £18.00
QL Psion Software Separate programs £10.00
QL power Supply Unit £10.00
TV or Network Leads £3.00
IC's ZX 8301 £6.00 ZX 8302 £3.00
MC 68008 £12.00
QL SERVICE MANUALS & CIRCUIT £25.00

Membrane (and instructions) £9.00
QL Top & Bottom Case £5.00
8049 (IPC) £3.00 MC 1377 £3.00

S.A.E. FOR FURTHER DETAILS



Payment terms: CWO, Access, VISA at cashiers Cheques - allow 10 days

Delivery: Carriage - £9 Postage - £5



Product subject to availability. E&OE TEL: 0753 888866

FAX: 0753 887149



DILWYN JONES COMPUTING

41 BRO EMRYS, TAL-Y-BONT, BANGOR,
GWYNEDD, LL57 3YT, GREAT BRITAIN

TELEPHONE: (0248) 354023

QL SOFTWARE

A SELECTION FROM OUR RANGE OF NEARLY 100 PRODUCTS FOR THE QL, NO
ROOM TO ADVERTISE THEM ALL HERE, SO PLEASE ASK FOR OUR
CATALOGUE (PHONE FOR A COPY OR SEND A LETTER WITH YOUR ADDRESS).

EASYPTR III part £40.50

Simplified pointer environment programming. Part I consists of sprite editor, menu editor and superbasic extensions to use in your own programs. Applications created using Easypr III can be compiled with QLiberator. Requires expanded memory, available on disk only.

EASYPTR III part 2 £20.00

Consists of appendix manager and enhanced toolkit for control of menus etc in your programs. EASYPTR III part 3 £20.00

Consists of Easysource and C library routines etc.

QLIBERATOR £50.00

Superb superbasic compiler, compiles virtually all of basic plus most toolkit commands, etc. Produce faster multitasking code from your basic programs. Compile resident extensions, use overlays etc. with the latest V3.36. Can be mouse controlled. Expanded memory required.

BUDGET QLIBERATOR £25.00

Excellent value, virtually all of superbasic but without some of the additional facilities of the full version. Not mouse controlled. Works on unexpanded QL too.

DJTOOLKIT £10.00

Compact toolkit of BASIC extensions, ideal for use with ALiberator. Really useful programming commands, can be distributed with your compiled programs if you wish. At this price, a bargain! Suitable for unexpanded QL.

LINEDESIGN £100.00

Vector drawing package, uses outline fonts and clipart, move and resize text and graphics without loss of quality. Ideal for making posters, etc. Supplied with huge range of fonts and clipart on TEN disks! The more memory your system has, the better! Available on disk only, can be mouse controlled (including SERmouse).

DATA DESIGN 3 £60.00

Superb, fast pointer driven database with free form field structures, with the option of disk based for large files if required, or smaller files can be kept in memory for speed. You do not have to be able to program this version but if you add the API package, it can be programmed from basic, C, or assembler. Expanded memory required, disk only.

API for Data Design £20.00

QPAC2 £39.95

Tony Tebb's superb pointer environment package. In addition to the pointer environment files themselves, this includes tutorials, extensive manual, files menu, channels and jobs menus, easy switching between jobs, hotkeys, etc. Mouse or keyboard controlled, a good introduction to pointer environment, 256k ram minimum, disk only

QPAC1 £19.95

Ideal companion to QPAC2, consists of small accessory programs such as calculator, calendar, clocks, alarm clocks, typewriter, etc. All can be mouse controlled. Pointer environment files included. Can be used with or without QPAC2. Expanded memory required, disk only.

QTPY2 £29.95

Tony Tebb's spelling checker program. Check spelling as you type OR check existing files retrospectively. User interface allows you to write programs which use the dictionary facilities. English, French and German dictionaries included!

DISA £29.00

Interactive pointer driven machine code disassembler. 256k ram min. Disk only.

MEGATOOLKIT £25.00

EPROM VERSION £40.00
Large toolkit with over 200 BASIC extensions, suitable for use with QLiberator or Turbo. Many examples supplied, extensive manual. Suitable for unexpanded QL.

DISCOVER £20.00

The painless way to move files from QL to PC and vice versa. As simple as copying files between two disks. 256k ram min., disk only.

MULTI DISCOVER £30.00

In addition to Discover facilities, also contains CPM, Unix CP10, BBC micro and now Spectrum and SAM Coupe file transfer capability. 256k min. ram, disk only.

TEXTIDY £15.00

Assists Discover with conversion of text files by stripping out control codes etc. 256k ram min.

CONVERT-PCX £10.00

Used with Discover, allows transfer of bit mapped PC clipart graphics in PCX format (a common PC file format) to QL screens or Page Designer pages. 256k ram, disk only.

QL-PC-FILESERVER £24.50

Link a PC and a QL via a serial port cable and use this software to enable the two to communicate - the QL can save its files on a PC's disk systems and print to the PC's serial port using normal basic commands like COPY. Works on unexpanded QL.

BANTER £25.00

Simple to use banner maker which uses outline fonts for good quality large texts. Prints sideways across up to 4 sheets of paper. Simple to use, menu driven, on screen preview before printing, etc. Suits most Epson compatible printers.

IMAGE PROCESSOR 2 £15.00

Easy to use graphics system, featuring usual graphic facilities, pixel zoom editing, image enhancement, mode conversion etc. 512k, disk only.

SCREEN COMPRESSION £10.00

Reduce the amount of storage required by graphics on disk or microdrive - supports several QL formats. 256k, disk only.

SCREEN DAZZLER £15.00

Unlike the usual screen savers, which simply turn off the display when the keyboard is not used for a while, this one can activate various graphical displays to provide an attractive means of preventing screen burn-in, more like the screen savers on other computers. If you have a compiler, you can even write your own savers by following the instructions in the manual. Pointer environment compatible.

SCANNED CLIPART 1 £10.00

NEW! A disk full of compressed scanned pictures (decompression program supplied of course) which can be used in most QL programs (DTP, graphics, etc). Assorted collection, containing many pictures you may not find in other collections. Large number of pictures, a bargain at this price. 128k, disk only.

PRINTERMASTER £20.00

Select printer control codes quickly and simply from a menu to set fonts, page lengths, etc. before printing from programs like Quill, etc. 128k, disk/mdv

SERMOUSE £40.00

Albin Hessler's serial mouse driver system for the QL is now available from DJC complete with a QL style matching black mouse with 9 pin serial connector and UK style port adaptor lead. Version 3 driver software. Can now work with The Painter too. NB POSTAGE CHARGES BELOW

MAGAZINES EX-CGH SERVICES

Ask for a price list of back issues of QL Technical Review, QL Leisure Review and QL Adventurer's Forum (all available at time of writing).

SQUIDQY ROUND THE WORLD £12.50

An arcade game, ideal for the young at heart! 128k

5-GAMES PACK £12.50

5 'thinking' games in one bargain pack, 128k

SUPPLIES

FLOPPY DISKS £0.40

DSHD DISKS £0.70

MICRODRIVES £2.50

DISK LABELS £2.00

On printer roll £2.50

ADDRESS LABELS £2.00

MDV LABELS £2.00

MOUSE MATS £2.50

Disk box dividers £3.00

in stock once more!

TERMS: Discounts - buy 2 programs, claim 5% off each, buy 3 or more, claim 10% off each program. Offer applies to software only. POSTAGE - Software is sent post free to UK addresses. Overseas please add £1.00 per program for postage (maximum £3.00). Floppy disks and serial mouse - add postage of £2.50. Labels/mouse mats - add postage of £0.50 per item if only buying these. PAYMENT - In UK currency (pounds sterling) only please. Payment by cheque, Eurocheque, Postal Order, cash (send by registered post), or by credit card (Visa/Mastercard/Eurocard/Connect). In case of difficulty contact us first to arrange a payment method if none of these is possible for you. Please make cheques, etc payable to DILWYN JONES COMPUTING (not to any other name or abbreviation please, our bank prefers it that way).